Université Fédérale

Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut Supérieur de l'Aéronautique et de l'Espace (ISAE)

Présentée et soutenue par :
Jonathan GUERRA

le jeudi 20 octobre 2016

Titre :

Optimisation multi-objectif sous incertitudes de phénomènes de thermique transitoire

---

École doctorale et discipline ou spécialité :

ED MITT : Mathématiques appliquées

Unité de recherche :

Équipe d'accueil ISAE-ONERA MOIS

Directeur(s) de Thèse :

M. Fabrice GAMBOA (directeur de thèse)
Mme Patricia KLOTZ (co-directrice de thèse)

Jury :

M. Marc SCHOENAUER, Directeur de recherche INRIA - Président du jury
M. Patrick CATTIAUX, Professeur d'Université
M. Sébastien DA VEIGA, Ingénieur SAFRAN
M. Nicolas DOLIN, Ingénieur Epsilon
M. Fabrice GAMBOA, Professeur d'Université - Directeur de thèse
Mme Patricia KLOTZ, Ingénieur de recherche ONERA - Co-directrice de thèse
M. Rodolphe LE RICHE, Directeur de recherche CNRS - Rapporteur
M. Luc PRONZATO, Directeur de recherche CNRS - Rapporteur

Résumé

La résolution d'un problème d'optimisation multi-objectif sous incertitudes en présence de simulations numériques coûteuses nécessite le développement de techniques parcimonieuses en optimisation et en propagation d'incertitudes. L'application visée ici porte sur des phénomènes de thermique transitoire. Dans ce cadre, nous développons également des modèles de substitution spatio-temporels.

Dans un premier temps, nous étudions un algorithme multi-objectif capable de restituer une discrétisation régulière du front de Pareto complet tout en nécessitant peu d'appels aux fonctions objectif. L'algorithme proposé est basé sur les processus Gaussiens et la méthode EGO. Pour que les solutions soient bien réparties sur le front de Pareto, la maximisation de l'amélioration espérée multi-objectif est effectuée en utilisant l'algorithme génétique NSGA-II, appliqué à la prédiction par Krigeage des objectifs. Ceci permet en particulier d'améliorer la capacité d'exploration de l'espace des entrées.

Le problème d'optimisation sous incertitudes est ensuite étudié en considérant des mesures de robustesse pire cas et probabilistes. L'optimisation pire cas considère la valeur maximum atteinte par la fonction objectif quelle que soit la valeur prise par les aléas. Des solutions moins conservatives peuvent être obtenues avec un critère probabiliste. Parmi ceux-ci, le superquantile, en intégrant tous les évènements pour lesquels la valeur de la sortie se trouve entre le quantile et le pire cas, fournit aussi une information sur le poids de la queue de distribution. Ces mesures de risque nécessitent un grand nombre d'appels à la fonction objectif incertaine. Ce nombre peut être réduit à l'aide d'un couplage avec l'algorithme multi-objectif, ce qui permet la réutilisation des évaluations déjà réalisées.

Peu de méthodes permettent de calculer le superquantile de la distribution de la sortie de fonctions coûteuses. Nous développons un estimateur du superquantile basé sur une méthode d'échantillonnage préférentiel et le Krigeage. Il permet d'approcher les superquantiles avec une faible erreur et une taille d'échantillon limitée. L'application de ces deux mesures de risque au cas industriel a mis en évidence qu'une solution Pareto-optimale pour le pire cas ne l'est pas forcément pour le superquantile et a conduit à des solutions innovantes et performantes.

Dans une dernière partie, nous construisons des modèles de substitution spatio-temporels, nécessaires dans le cas où le temps d'exécution de la fonction objectif incertaine est trop grand. Afin de répondre au cadre imposé par l'applicatif industriel de thermique transitoire, le modèle de substitution doit être capable de prédire des phénomènes dynamiques non-linéaires sur des temps longs et avec peu de trajectoires d'apprentissage. Les réseaux de neurones récurrents sont utilisés et une méthodologie de construction facilitant l'apprentissage est mise en place. Elle est basée sur la validation croisée et un calcul des poids du réseau réalisé par une optimisation multi-niveaux. Des techniques d'analyse de sensibilité sont également appliquées pour diminuer la dimension d'entrée du réseau.

Mots clés : optimisation multi-objectif, propagation d'incertitudes, optimisation sous incertitudes, krigeage, modèle de substitution spatio-temporel, réseaux de neurones récurrents.

Abstract

The resolution of multi-objective problems under uncertainties in the presence of costly numerical simulations requires the development of parsimonious optimization and uncertainty propagation techniques. Because the aimed application is a transient thermal engineering one, we have also developed spatio-temporal surrogate models.

First of all, we study a multi-objective optimization algorithm capable of returning a complete Pareto front by only using few calls to the objective functions. The proposed algorithm is based on Gaussian processes and on the EGO method. In order to return a uniformly discretized Pareto front, the maximization of the multi-objective expected improvement is processed thanks to the NSGA-II genetic algorithm, applied on the Kriging prediction of the objectives. In particular, this allows to improve the input space exploration ability of the method.

The optimization problem under uncertainties is then studied by considering the worst case and a probabilistic robustness measure. The worst case strategy considers the maximum of the value attained by the objective function, no matter the value taken by the uncertainties. Less conservative solutions can be found with a probabilistic criterion. Among these, the superquantile also gives an information on the weight of the distribution tail by integrating every events on which the output value is between the quantile and the worst case. Those risk measures require an important number of calls to the uncertain objective function. This number can be reduced thanks to a coupling with the multi-objective algorithm which enables to reuse previously computed evaluations.

Few methods give the possibility to approach the superquantile of the output distribution of costly functions. To this end, we have developed a superquantile estimator based on the importance sampling method and Kriging surrogate models. It enables to approach superquantiles with few error and using a limited number of samples. The application of those two risk measures on the industrial test case has highlighted that a Pareto optimal solution for the worst case may not be Pareto optimal for the superquantile. It has also led to innovative and competitive solutions.

In the last part, we build spatio-temporal surrogate models. They are necessary when the execution time of the uncertain objective function is too large. In order to answer to the framework imposed by the transient thermal industrial application, the surrogate model has to be able to predict dynamic, long-term in time and non linear phenomena with few learning trajectories. Recurrent neural network are used and a construction facilitating the learning is implemented. It is based on cross validation and on a weights computation performed by multi-level optimization. Sensitivity analysis techniques are also applied to lower the input dimension of the neural network.

Keywords : multi-objective optimization, uncertainty propagation, optimization under uncertainties, spatio-temporal surrogate models, Kriging, importance sampling.

Remerciements

Ces trois années de thèse m'ont grandi, tant personnellement que professionnellement. Je tiens donc à remercier toutes les personnes qui ont rendu ce travail possible dans des conditions idéales.

Je remercie d'abord, avec toute mon amitié, mes deux encadrants de thèse Madame Patricia Klotz et Monsieur Fabrice Gamboa. Patricia, merci pour les échanges scientifiques que l'on a eu et qui ont permis de faire mûrir mes recherches en m'aidant continuellement à prendre du recul. Merci également pour tes conseils, ces derniers ayant largement bonifié ce travail. Ce fut un réel plaisir de partager ce bureau avec toi, j'espère que l'on aura l'occasion de continuer à collaborer ensemble sur différents projets à l'avenir.

Fabrice, merci d'avoir quotidiennement trouvé du temps entre deux déplacements pour m'accueillir dans la bonne humeur et discuter de l'avancement de mes travaux. Tes connaissances dans de nombreux domaines des mathématiques ont énormément contribué à la diversité du travail réalisé dans cette thèse. Merci aussi pour ta bienveillance malgré les lacunes que j'avais en probabilité et statistique. Ta patience et ta pédagogie ont participé à les combler.

Je remercie ensuite grandement Messieurs Luc Pronzato et Rodolphe Le Riche, non seulement d'avoir accepté de rapporter mon mémoire de thèse mais aussi pour la précision avec laquelle leur relecture a été réalisée. Vos retours m'ont véritablement aidé à améliorer la qualité de mon manuscrit.

Je tiens aussi à remercier les membres du jury, Messieurs Marc Schoenauer et Sébastien Da Veiga pour l'honneur qu'ils m'ont fait de s'intéresser à mon étude et la pertinence des questions qu'ils m'ont posées le jour de ma soutenance. Tout cela a contribué à enrichir mon questionnement et donc mon mémoire.

Ce travail n'aurait pas été possible sans le financement de la société Epsilon ingénierie. Pour cette raison, ainsi que pour sa confiance et son indéfectible soutien, je remercie Nicolas Dolin, directeur technique de l'entreprise. Je lui suis également très reconnaissant pour la qualité de son suivi de mes travaux malgré le peu de temps disponible pour le faire. Les questions pertinentes qu'il m'a posées tout au long de la thèse ont, de plus, largement favorisé la définition et la résolution du problème industriel.

Je souhaite également remercier Monsieur Patrick Cattiaux et Madame Béatrice Laurent. L'attention et l'intérêt qu'ils ont portés à mes recherches ont donné lieu à des réunions plaisantes et constructives qui ont alimenté mon étude. Il me faut plus particulièrement dire merci à Patrick pour ses cours de statistiques qui ont contribué à compenser un certain nombre de mes manques.

Je tiens aussi à remercier Emmanuel Laroche, ingénieur de recherche à l'ONERA, pour le suivi de mes travaux. Son expertise thermique a concouru à faire le pont entre les problématiques thermiques industrielles et le milieu de la recherche académique.

Pour leur accueil dans le département et la bonne ambiance qu'ils perpétuent dans les locaux de l'ONERA, je remercie les différents permanents : Vincent Mouysset pour nos discussions sur le sport, le cinéma... et pour les rires que nous avons pu partager, Jean-Lou Bussenot pour sa réactivité au moindre problème informatique et pour son amicalité ou encore à tous ceux avec qui j'ai partagé des séances de futsal le midi. Je souhaite également remercier François Rogier de m'avoir accueilli au sein du département M2SN.

J'aimerais dire merci au projet européen TOICA, et en particulier à Fabien Mangeant, qui m'a donné l'occasion, et ce à plusieurs reprises, de mettre en valeur mes travaux au

sein du projet. Il m'a ainsi offert la possibilité de confronter mes méthodes aux divers besoins existants dans l'industrie.

À tous les ingénieurs d'Epsilon, merci de m'avoir aidé à comprendre et à intégrer dans ma démarche les enjeux particuliers à la thermique. Je tiens d'abord à remercier Sébastien d'avoir continuellement fait le lien entre mes travaux et le projet européen, ensuite Laure d'avoir pris le temps de lancer des calculs 3D thermique pour la validation de la construction de réseaux de neurones développée dans cette thèse et puis Xavier pour son assistance sur les problèmes de thermomécanique du cas industriel et pour nos nombreux échanges scientifiques. Je suis également reconnaissant à Steve, Arnaud, Alexis, Lola, Laurent, Yoyo, Guigui ... pour nos diverses conversations lors des repas du midi, les 'émulations' du vendredi, les futsals et autres tournois de foot.

Aux doctorants et stagiaires de l'ONERA, Marc, Alexandra, Asma, Matthieu, Emmanuele, Sarra, Matthias, Tomasz, Dérek ... merci pour les pauses-cafés que nous avons partagées et pour les discussions diverses et variées que nous avons pu y avoir.

À mon entourage et à mes amis proches, Jean-Thomas, Yannick, Hamid, Hadrien, Lisa, et tous les autres, je tiens à vous dire merci de m'avoir donné la possibilité de penser à autre chose qu'à ma thèse et ce en toute circonstance.

Avec toute mon affection, je remercie ma belle-famille et plus particulièrement mes beaux-parents, Claude et Jean-Marie, pour leur soutien et pour leur générosité.

Je suis aussi infiniment reconnaissant à ma famille, et surtout à mes parents, Alain et Henriette, de m'avoir toujours poussé et assisté dans la réalisation de mes études. Je n'aurais jamais pu aller aussi loin sans vous.

Enfin, je remercie de tout mon cœur ma femme, Sarah, pour m'avoir toujours suivi et soutenu dans mes choix, quels qu'ils soient, et pour le réconfort qu'elle a su m'apporter tout au long de cette thèse.

Table des matières

Première partie

Introduction

Chapitre 1

Introduction générale et plan du manuscrit

Dans le but de réduire la consommation en carburant des avions de prochaine génération, l'une des tendances actuelles est d'augmenter la part d'électrique dans leur fonctionnement, jusqu'à envisager des avions entièrement électriques. Pour cela, il est nécessaire de mettre en place davantage d'équipements électroniques dans un avion, sachant qu'ils sont jusqu'à présent principalement placés dans la baie avionique qui est un compartiment thermiquement régulé situé sous le cockpit. Or, la concentration de nombreux équipements induit un échauffement des composants électroniques et donc une possible diminution de leur durée de vie. Pour pallier cela, l'une des solutions possibles est de trouver de nouveaux emplacements où positionner ces meubles électroniques.

Toutefois, ces nouveaux emplacements ont le désavantage de ne pas être systématiquement thermiquement régulés. Ceci oblige l'architecte à jouer sur les solutions de refroidissement afin d'éviter la surchauffe et de garantir une durée de vie équivalente à celle que l'équipement avait dans la baie avionique. En outre, modifier les solutions de refroidissement n'a pas uniquement un impact sur la durée de vie mais également sur la masse et le coût de production, qu'il est indispensable de minimiser afin que l'architecture proposée soit concurrentielle.

Par ailleurs, afin d'estimer la durée de vie de l'équipement, il est nécessaire de modéliser l'évolution au cours du temps de la température de la structure de l'équipement. Pour cela, il est indispensable de connaître l'évolution au cours d'un vol des propriétés physiques environnementales telles que la température de l'air et des structures dans le nouvel emplacement. Cependant, cet emplacement n'étant initialement pas prévu pour accueillir l'équipement, ses propriétés physiques ne sont pas connues de manière certaine. Cela implique que la durée de vie de l'équipement est également incertaine. Afin d'ajuster au mieux l'équipement, il est donc nécessaire d'être capable d'estimer la pire valeur de durée de vie quelles que soient les incertitudes sur les solutions de refroidissement et sur les conditions environnementales. La problématique industrielle, ici introduite succinctement, est plus amplement détaillée au chapitre 2.

Enfin, un problème de conception peut se traduire en un problème d'optimisation. L'optimisation numérique consiste à trouver numériquement le minimum de fonctions objectif en modifiant des variables de contrôle qui influent sur ces objectifs lorsqu'ils sont modifiés. Dans le cas de l'équipement, il s'agit de minimiser la masse, le coût et de maximiser la durée de vie quelles que soient les incertitudes. Les variables de contrôle sont ici des traductions mathématiques de solutions de refroidissement.

Pour résumer, le déplacement d'équipements électroniques de la baie avionique vers de nouveaux emplacements, comme d'autres problèmes en ingénierie, peut se ramener à la résolution d'un problème d'optimisation multi-objectif sous incertitudes. Plus précisément, le pire cas, qui est la pire valeur prise par les fonctions objectif quelles que soient les incertitudes, doit être minimisé. Par ailleurs, les modèles permettant de calculer la température d'un équipement au cours du temps pouvant nécessiter plusieurs heures, la résolution de ce problème d'optimisation sous incertitudes doit être parcimonieuse, c'est-à-dire nécessiter un nombre réduit d'appel aux fonctions objectif pour atteindre la convergence. Pour répondre à cela, cette étude s'intéresse donc à la résolution d'un problème d'optimisation multi-objectif sous incertitudes en présence de simulations numériques coûteuses.

Cette résolution nécessite le développement de techniques parcimonieuses en optimisation et en propagation d'incertitudes. Pour cela, l'utilisation de modèles de substitution est indispensable. Ce sont des modèles analytiques à temps de réponse rapide qui se substituent au modèle de référence. Des exemples de modèles de substitution usuels sont les réseaux de neurones artificiels [Dreyfus *et al.*, 2002] ou les processus gaussiens aussi appelés krigeage [Krige, 1951]. Ces modèles analytiques sont ajustés au modèle de référence à l'aide de simulations provenant de ce dernier et qui forment un plan d'expériences. Une fois ce plan d'expériences fourni, la construction des modèles de substitution repose sur l'apprentissage statistique, développé notamment dans [Hastie *et al.*, 2009]. Cette construction se décompose en plusieurs étapes : d'abord le modèle de substitution doit être ajusté au modèle de référence par minimisation de l'erreur commise sur le plan d'expériences. Le modèle de substitution étant paramétré, ce sont ces paramètres qui servent à minimiser l'erreur commise. Ensuite, le nombre de ces paramètres étant variable, il est nécessaire de déterminer la complexité optimale du modèle de substitution. Cela consiste à rechercher le nombre de paramètres du modèle analytique qui minimise l'erreur de généralisation. Par exemple, la validation croisée [Efron, 1983] ou le *bootstrap* [Efron, 1979] sont deux techniques permettant d'évaluer la capacité de généralisation d'un modèle en présence d'un plan d'expériences de taille réduite. Dans le cas où le modèle de référence correspond à des simulations instationnaires d'un phénomène physique, il est également nécessaire de s'intéresser aux techniques de construction de modèle de substitution spatio-temporels existantes, par exemple basées sur l'identification de modèle [Ljung, 1999]. Le chapitre 3 présente donc les techniques d'apprentissage statistique pour la construction de modèles de substitution statiques et spatio-temporels.

Les variables de contrôle ainsi que certains paramètres environnementaux du problème d'optimisation à résoudre sont incertains. Puisque les ingénieurs ne veulent pas sous-estimer les effets des aléas sur les fonctions objectif, nous nous intéressons plus précisément aux mesures de risque permettant de quantifier les effets les plus extrêmes des incertitudes. Rockafellar en définit plusieurs dans son étude [Rockafellar, 2007], et la manière de les estimer. Parmi celles-ci, le superquantile, aussi appelé *Conditional Value at Risk*, en intégrant tous les évènements pour lesquels la valeur de la sortie se trouve entre le quantile et le pire cas, fournit aussi une information sur le poids de la queue de distribution. Puisque cette mesure de risque nécessite un grand nombre d'appels à la fonction objectif incertaine, les techniques permettant de réduire la variance de l'estimateur Monte-Carlo avec des échantillons de taille inférieure sont détaillées, telles que l'échantillonnage préférentiel ou l'échantillonnage stratifié, provenant par exemple du livre [Morio et Balesdent, 2015]. L'assistance de ces méthodes par des modèles de substitution de type krigeage [Balesdent *et al.*, 2013] [Chevalier *et al.*, 2014] permet de plus de diminuer le nombre d'appels à la fonction ob-

jectif incertaine. Toutefois, ces techniques ne concernent que l'estimation du quantile et leur application au superquantile n'est pas directe. Le chapitre 4 dresse un état de l'art de ces techniques de propagation des incertitudes à travers un modèle déterministe pour l'estimation de mesures de risque.

Le problème industriel est un problème d'optimisation multi-objectif. Les ingénieurs utilisent l'optimisation afin de trouver des solutions qu'ils n'auraient pas pu obtenir par l'expertise. Ils sont donc intéressés par l'ensemble des compromis optimaux entre les différents critères, aussi appelé front de Pareto. Le front de Pareto correspond donc à un ensemble de solutions innovantes. De ce fait, nous sommes intéressés par les algorithmes permettant de retourner une discrétisation régulière du front de Pareto complet. De plus, le temps de restitution devant être minimisé, il est donc nécessaire de faire un état de l'art du domaine et de s'intéresser plus particulièrement aux algorithmes parcimonieux en nombre d'appels aux fonctions objectif ou permettant de distribuer les appels à ces dernières. L'algorithme appelé EGO [Jones *et al.*, 1998] est un bon exemple d'algorithme parcimonieux. Il consiste à utiliser l'erreur d'estimation fournie par le krigeage afin d'enrichir séquentiellement le plan d'expériences avec le point où le minimum a le plus de probabilité de se réaliser. Ceci se fait en maximisant un critère appelé l'amélioration espérée (ou Expected Improvement EI). Cet algorithme a permis de réduire drastiquement le nombre d'appels nécessaires à la fonction à minimiser afin de converger vers la solution. Par contre, il ne permet de minimiser qu'un seul objectif. Cet algorithme a depuis été généralisé par d'autres auteurs à plusieurs objectifs. Par exemple, Forrester et Keane [Forrester et Keane, 2009] ou encore Emmerich [Emmerich *et al.*, 2011] l'ont fait en adaptant le critère EI à plusieurs objectifs en retraduisant ce qu'est une amélioration espérée en la présence de plusieurs objectifs. Cela permet, dans le cas où le nombre de paramètres de contrôle est réduit, d'obtenir un front en un nombre d'itérations restreint. Cette méthode a en revanche deux limites : d'abord, la généralisation de l'EI ne garantit pas une capacité d'exploration du Front de Pareto suffisante. La deuxième est que ce critère est optimal pour un enrichissement à un pas. Or, il est indispensable, vu le coût induit par les incertitudes de proposer des méthodes où l'on peut rajouter plusieurs points à chaque itération. Les algorithmes évolutionnaires tels que NSGA-II [Deb *et al.*, 2002] permettent justement d'explorer intensivement l'espace des paramètres de contrôle ce qui leur permet de résoudre un grand nombre de problème d'optimisation, quelles que soient le nombre de paramètres de contrôle, la régularité des fonctions objectif, ou la forme du front de Pareto à approximer. Enfin, le problème d'optimisation à résoudre nécessitant d'optimiser un pire cas, nous nous intéressons aux algorithmes résolvant les problèmes de type MinMax, telles que l'étude de Marzat [Marzat *et al.*, 2012] ou encore de Rehman [Rehman *et al.*, 2014]. Ces études proposent notamment de coupler l'algorithme d'optimisation et l'estimation de la mesure de robustesse pire cas, en présence d'un seul objectif. Le domaine de l'optimisation mono-objectif, multi-objectif et sous incertitudes est présenté au chapitre 5.

Pour résumer, il existe des algorithmes mono-objectifs et multi-objectifs efficaces pour résoudre des problèmes incertains. Néanmoins, aucun ne permet de retourner en un temps raisonnable un front de Pareto optimal et bien discrétisé pour des mesures de robustesse de type pire cas ou superquantile. En effet, ces mesures de risque sont peu utilisées en optimisation du fait du nombre élevé d'appels à la fonction objectif qu'elles nécessitent, le nombre d'évaluations nécessaires de ces mesures de risque étant d'autant plus important en présence de plusieurs objectifs. De plus, le superquantile est une mesure peu répandue en ingénierie et il en existe de ce fait peu d'estimateurs parcimonieux. Or, elles corres-

pondent souvent à la demande des ingénieurs et l'évaluation des fonctions objectif peut être coûteuse. L'objectif de cette thèse est donc de proposer un algorithme d'optimisation multi-objectif pire cas et superquantile en présence de fonctions objectif coûteuses à évaluer. L'application visée ici porte sur des phénomènes de thermique transitoire. Dans ce cadre, nous développons également des modèles de substitution spatio-temporels permettant de réduire davantage, si nécessaire, le nombre d'appels aux simulations numériques coûteuses.

Notre démarche a consisté, dans un premier temps, à développer un algorithme multi-objectif capable de restituer une discrétisation régulière du front de Pareto complet tout en nécessitant peu d'appels aux fonctions objectif. L'algorithme proposé est basé sur les processus gaussiens et la méthode EGO. Pour que les solutions soient bien réparties sur le front de Pareto, la maximisation de l'amélioration espérée multi-objectif est effectuée en utilisant l'algorithme génétique NSGA-II, appliqué à la prédiction par krigeage des objectifs. Ceci permet en particulier d'améliorer la capacité d'exploration de l'espace des entrées. Le chapitre 6 détaille cet algorithme et le compare à des méthodes existantes.

Le problème d'optimisation sous incertitudes est ensuite étudié en considérant des mesures de robustesse pire cas et probabilistes de type superquantile. Afin d'appliquer l'algorithme d'optimisation précédent à la sortie de ces deux mesures de risque, nous montrons d'abord la continuité, par rapport aux paramètres d'optimisation, de l'application de la mesure de risque sur une fonction objectif continue. En outre, ces mesures de risque nécessitent un grand nombre d'appels à la fonction objectif incertaine. Ce nombre peut être réduit à l'aide d'un couplage avec l'algorithme multi-objectif, ce qui permet la réutilisation des évaluations déjà réalisées. Le chapitre 7 détaille ce point de vue ainsi que l'application de l'algorithme multi-objectif pire cas au problème industriel.

Pour l'évaluation du superquantile, peu de méthodes existent pour l'estimer sur la distribution de la sortie de fonctions coûteuses. Nous développons donc au chapitre 8 un estimateur du superquantile basé sur une méthode d'échantillonnage préférentiel et le krigeage. Il permet de l'approcher avec une faible erreur et une taille d'échantillon limitée. En modifiant l'algorithme multi-objectif pire cas précédent en remplaçant uniquement la mesure de robustesse, nous obtenons l'algorithme multi-objectif probabiliste. Ce dernier est appliqué au cas industriel. Les solutions obtenues par superquantile et par pire cas peuvent de ce fait être comparées afin de déterminer les avantages et les inconvénients de ces deux approches.

Au chapitre 9, nous construisons des modèles de substitution spatio-temporels, nécessaires dans le cas où le temps d'exécution de la fonction objectif incertaine est trop grand. Afin de répondre au cadre imposé par l'applicatif industriel de thermique transitoire, le modèle de substitution doit être capable de prédire des phénomènes dynamiques non-linéaires sur des temps longs et avec peu de trajectoires d'apprentissage. Les réseaux de neurones récurrents sont utilisés et une méthodologie de construction facilitant l'apprentissage est mise en place. Elle est basée sur la validation croisée et un calcul des poids du réseau réalisé par une optimisation multi-niveaux. Des techniques d'analyse de sensibilité sont également appliquées pour diminuer la dimension d'entrée du réseau. Ce chapitre a d'ailleurs fait l'objet d'une publication [Guerra *et al.*, 2015].

Enfin, le chapitre 10 étudie le comportement asymptotique théorique des estimateurs par échantillonnage préférentiel des mesures de risque que sont le quantile et le superquantile. Plus précisément, l'évolution du comportement de ces estimateurs en différents jeux de paramètre de contrôle (cadre imposé par l'optimisation) est étudiée.

Chapitre 2

Problématique Industrielle - Thermique des équipements électroniques dans un avion

Sommaire

2.1 Contexte général de l'étude

Dans le cadre de l'étude d'un équipement électrique, le principal objectif d'un ingénieur thermicien est de garantir que les composants électroniques permettant le fonctionnement de cet équipement n'atteignent pas des températures trop élevées. En effet, une surchauffe trop fréquente de ces composants endommage leur structure, ce qui peut mener à leur rupture et donc à une fin de vie prématurée. En d'autres termes, plus la température de la structure des composants électroniques de l'équipement est élevée, plus la durée de vie de l'équipement peut être réduite.

Afin de ne pas avoir de problèmes liés à cette surchauffe, les équipements sont jusqu'à présent principalement placés dans la baie avionique, située en dessous du cockpit. Du fait que cet emplacement a une température ambiante régulée par la climatisation, l'étude du comportement thermique des équipements électroniques dans les avions n'était

pas une priorité pour les avionneurs et était de ce fait assez limitée : elle se réduisait à fournir un cahier des charges aux équipementiers avec une température à ne pas dépasser par l'équipement en fonctionnement nominal dans un environnement idéal, c'est-à-dire à température ambiante constante, sans considérer les interactions avec d'éventuels voisins.

Afin de réduire la consommation des avions de prochaine génération, l'une des tendances actuelles est d'augmenter la part d'électrique dans leur fonctionnement, jusqu'à envisager des avions entièrement électriques. L'une des conséquences est l'augmentation du nombre d'équipements électroniques à positionner. Il devient impossible de placer tous ces nouveaux équipements dans la baie avionique et il est donc nécessaire de trouver de nouveaux endroits où les placer. On peut par exemple penser à une cavité proche des ailes, ce qui aurait l'avantage de diminuer la longueur des câbles permettant de les relier aux génératrices de courant provenant des moteurs, et donc de réduire la masse induite par l'ensemble. L'une des limites à cela réside dans le fait que les nouvelles positions envisagées ne sont pas forcément climatisées ou pressurisées comme la baie avionique, ce qui implique que la température ambiante peut être élevée. Ceci signifie que les équipements électroniques placés dans cette cavité surchauffent davantage par rapport à un équipement placé dans la baie avionique et vont donc avoir une durée de vie plus réduite.

Le besoin de trouver de nouveaux emplacements combiné au fait que l'environnement de ces derniers peut favoriser la surchauffe des équipements imposent de travailler sur l'architecture thermique de ces équipements de manière différente et ce dès les phases d'avant-projet. Plus précisément, le thermicien doit intégrer les solutions de refroidissement de l'équipement dès sa phase de conception de sorte à augmenter les échanges de chaleur avec l'environnement extérieur. L'objectif de cette modification est d'assurer à l'équipement une durée de vie supérieure ou égale à celle qu'il avait dans la baie avionique. Toutefois, ce contrôle de la durée de vie ne doit pas se faire au détriment de la masse et du surcoût qu'impliqueraient ces modifications. En effet, la solution proposée serait inacceptable si elle induisait une augmentation trop importante de l'un de ces deux critères.

Par ailleurs, du fait du positionnement amont de cette étude et de la présence de différents acteurs (équipementier, avionneur,...), il existe de nombreuses sources d'incertitude. On peut par exemple citer les conditions environnementales dans la nouvelle position, les incertitudes liées aux modèles eux-mêmes, ou encore la connaissance imparfaite des paramètres des solutions de refroidissement envisagées (dues à des contraintes d'usinage par exemple). Il est donc crucial de quantifier toutes ces incertitudes de manière à les prendre en compte dans la décision de déplacement de l'équipement.

Afin de traduire ce problème dans le formalisme de l'optimisation numérique, les thermiciens doivent donc :

— Modéliser la température d'un équipement, son effet sur la durée de vie et choisir une manière de quantifier les incertitudes sur les paramètres d'entrée.
— Définir l'équipement étudié, les solutions de refroidissement envisagées et leur traduction en paramètres d'optimisation. Enfin, définir les incertitudes : sur quels paramètres vont-elles porter ? De quel type sont-elles ? Quelle en est leur amplitude ?
— En tirer le problème d'optimisation à résoudre.

2.2 Modélisation

2.2.1 Modélisation thermique

Afin d'évaluer la surchauffe des composants de l'équipement électronique, il faut être capable d'estimer la température à l'intérieur de l'équipement. On cherche donc un champ de température $T(\boldsymbol{p}, t)$, avec $\boldsymbol{p} \in \mathbb{R}^3$ la position spatiale et $t \in \mathbb{R}$ le temps. Ce champ de température est solution de l'équation de la chaleur suivante :

$$\rho c \frac{\partial T(\boldsymbol{p}, t)}{\partial t} = \operatorname{div}\left(\phi(\boldsymbol{p}, t)\right) + \Phi_v(\boldsymbol{p}, t) \tag{2.1}$$

avec ρ la masse volumique du matériau, c sa chaleur spécifique, λ la conductivité thermique du matériau (provenant de la loi de Fourier), ϕ la densité de flux de chaleur et Φ_v étant le terme forçant (en d'autres termes la densité volumique de puissance à dissiper par le système).

Afin de résoudre l'équation (2.1), on peut discrétiser le système étudié. Cette discrétisation peut se faire de différentes manières, selon la méthode de résolution utilisée. En thermique il existe : des logiciels utilisant la méthode des volumes finis (FLOTHERM, Fluent, ..), d'autres utilisant la méthode des éléments de frontière (EPSILON R3D) ou encore des logiciels basés sur la méthode des éléments finis tels que IDEAS-TMG.

2.2.1.1 Résolution de l'équation de la chaleur par modèle nodal

Une autre modélisation classiquement utilisée en thermique est la modélisation nodale, basée quant à elle sur les différences finies. Elle consiste en un découpage en éléments de volume $\mathcal{V}_i$ avec $i \in \{1...N_{el}\}$ et N_{el} le nombre de volumes utilisés, supposés isothermes. Ceci permet donc de connaître le champ de température en différents points $\boldsymbol{p}_i$ situés aux nœuds du réseau. Les équations sont issues d'une approche physique directe appliquée à un découpage du milieu en éléments de volume. Chaque volume $\mathcal{V}_i$ est défini par :

- son volume V_i
- sa masse volumique ρ_i
- sa chaleur massique C_{p_i}
- sa capacité thermique (inertie) $\rho_i C_{p_i} V_i$
- sa source de chaleur Φ_i
- sa température qui est celle du nœud $T_i = T(\boldsymbol{p}_i, t)$

En intégrant l'équation (2.1) sur chacun de ces volumes, le théorème de Green-Ostrogradski permet de transformer le terme correspondant à la divergence de la densité de flux de chaleur en un bilan de flux de chaleur entrant et sortant par la frontière de ce volume. Deux éléments $\mathcal{V}_i$ et $\mathcal{V}_j$ $(i \neq j)$ échangent le flux ϕ_{ij} selon différentes lois de transfert :

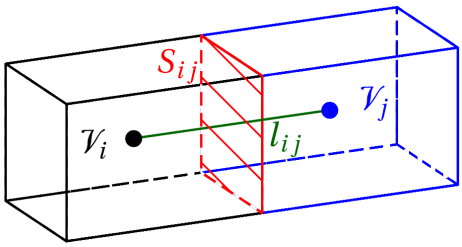


FIGURE 2.1 – Schéma représentant le découpage en volume d'un système

$$\phi_{ij} = C_{ij}(T_i - T_j) \tag{2.2}$$

avec C_{ij} la conductance thermique. Cette conductance varie selon le type d'échange thermique qu'elle représente. Elle dépend de la surface d'échange entre deux volumes S_{ij}. On peut distinguer trois types de transfert de chaleur :

1. La conduction, qui caractérise le transfert de chaleur dans la matière dû à un gradient de température, sans transport de matières. Cet échange se fait donc par contact entre deux volumes, ou au sein du volume, et le milieu peut être solide, liquide ou gazeux, en mouvement ou inerte. Dans le cas d'un flux conductif axial, la conductance associée se calcule de la manière suivante :

$$C_{ij} = \frac{\lambda S_{ij}}{l_{ij}} \tag{2.3}$$

   avec l_{ij} l'épaisseur le long de laquelle cette conduction se réalise (cf figure 2.1). Pour simplifier la formule, un seul λ est considéré mais si l'échange se fait entre deux matériaux différents, il faudrait séparer la composante de conduction dans le premier matériau défini par le couple (l_i, λ_i) puis la composante dans le second matériau (l_j, λ_j) et sommer les deux pour obtenir le C_{ij} total.

2. La convection, qui est un phénomène de transfert de chaleur avec transfert de matière. Ce phénomène se produit à l'interface de deux systèmes de phases différentes, par exemple solide-liquide ou solide-gaz. La conductance associée est :

$$C_{ij} = h(\mathcal{V}_i, \mathcal{V}_j)S_{ij} \tag{2.4}$$

   h étant le coefficient d'échange convectif. Ce dernier dépend des caractéristiques physiques du volume fluide (liquide ou gazeux). Par exemple, dans le cas d'une interface solide-gaz, ce h peut être calculé par des lois de paroi et dépend donc de la température et de la pression de l'air à proximité de la paroi.

   Dans le cas d'un fluide en mouvement, la chaleur peut également être transportée au sein même de ce fluide et l'on parle alors d'advection. À la différence des autres transferts de chaleur, celui-ci n'est pas réciproque, c'est-à-dire qu'il n'y a pas d'échange de chaleur entre deux volumes d'un même fluide : la chaleur passe d'un volume à un autre dans le sens du mouvement du fluide. La conductance associée est la suivante :

$$C_{ij} = \dot{m}C_{p_i} \tag{2.5}$$

   où $\dot{m}$ est le débit massique exprimé en kg/s et C_{p_i} la capacité thermique massique du fluide (aussi appelée chaleur massique).

3. Le rayonnement, est l'ensemble des échanges d'énergie à distance entre les corps par ondes électromagnétiques. Il ne nécessite pas de support matériel pour se propager et représente la seule possibilité d'échange thermique entre des corps distants placés dans le vide. En aéronautique et particulièrement pour l'étude de la thermique des équipements, sa contribution est souvent négligeable comparée à celle des deux autres types. L'échange de flux par rayonnement entre deux volumes ϕ_{ij} ne peut pas se mettre sous la forme définie par l'équation (2.2). Il est non-linéaire (dépendance en la température à la puissance quatre) et est souvent calculé de la manière suivante :

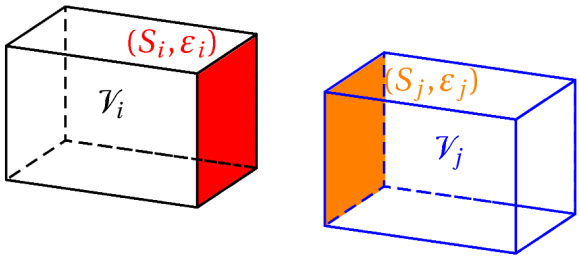


FIGURE 2.2 – Illustration des échanges par rayonnement de deux éléments de volume sans contact

$$\phi_{ij} = \Gamma_{ij}(T_i - T_j)^4 \tag{2.6}$$

Avec Γ_{ij} dépendant de la constante de Boltzmann, des émissivités ε_i, ε_j de chacune des deux surfaces concernées S_i et S_j (cf figure 2.2), de la surface S_j émettant le rayonnement et du facteur de vue entre ces deux surfaces. Ce facteur représente la fraction du flux total émis par la surface S_j qui atteint S_i.

Ainsi, pour approximer le champ de température, on intègre l'équation (2.1) sur le volume de chacun des N_{el} éléments de la discrétisation. Ceci fournit un système d'équations différentielles ordinaires non-linéaires à résoudre. Grâce au théorème de Green-Ostrogradski, le terme correspondant à l'intégrale de la divergence du flux dans le volume de l'équation (2.1) se ramène à la réalisation, pour chaque élément $\mathcal{V}_i$, du bilan de tous les flux de chaleur échangés avec les autres éléments. Grâce aux équations (2.2), (2.3), (2.4) et 2.6, on obtient donc pour chaque élément i :

$$\rho_i C_{p_i} V_i \frac{dT_i}{dt} = \sum_{j \neq i} C_{ij}^{cond}(T_i - T_j) + \sum_{j \neq i} C_{ij}^{conv}(T_i - T_j) + \sum_{j \neq i} \phi_{ij}^{ray} + \Phi_i \tag{2.7}$$

Les termes en rouge représentent les termes pouvant, dans le cas général, apporter de la non-linéarité à ce système d'équations.

Remarques sur la convection : on peut distinguer deux types de convection.

1. La convection forcée qui se produit lorsque le mouvement du fluide est imposé par une action extérieure comme l'utilisation d'un ventilateur ou d'une pompe.
2. La convection naturelle qui apparaît spontanément dans un fluide au sein duquel il existe un gradient de température. Le mouvement du fluide résulte de la différence

de densité entres les zones chaudes et zones froides de ce dernier. Cette convection est très difficile à calculer du fait de sa grande dépendance aux conditions aux limites. Cette caractéristique rend le calcul basé sur cette convection fortement non-linéaire et le passage du régime forcé au régime naturel est très difficile à gérer dans le cadre de cette modélisation. Il consiste en effet à passer d'un mode où les conditions aux limites sont maîtrisées puisque fournies (régime forcé) à un mode où elles sont inconnues (devant être calculées) alors que le résultat en est fortement dépendant. Enfin, les deux régimes ont des pas de temps caractéristiques très différents. En effet, le régime naturel est un phénomène dont la mise en place est lente et de ce fait, il nécessite un pas de temps réduit, ce qui induit un coût de calcul d'autant plus élevé.

2.2.1.2 Modèles réduits physiques

Intérêts des méthodes de réduction Les modèles définis précédemment nécessitent le plus souvent une discrétisation très fine pour donner des résultats satisfaisants, ce qui implique des temps de calcul rédhibitoires pour certaines applications. En thermique, il existe notamment deux cas où ce temps de restitution est crucial :

1. **Modélisation multi-niveaux / multi-échelles / multi-physiques**. Cette modélisation consiste à intégrer un ou plusieurs modèles dans un modèle d'ensemble : cela permet de modifier les paramètres importants des différents sous-ensembles tout en gardant un modèle global aisément exploitable. Cela rend également possible la mesure de l'influence mutuelle des sous-ensembles, comme l'impact d'une physique sur une autre, et leur contribution sur le comportement global.

   Afin de construire un modèle global s'exécutant en un temps acceptable, il est inévitable d'utiliser des modèles réduits pour restituer le comportement des sous-ensembles. Grâce à cela, on peut aussi prendre en compte un nombre de paramètres plus conséquents pour chaque physique/échelle/niveau sans trop augmenter la complexité du modèle global.

   Par exemple, les techniques multi-niveaux en thermique peuvent permettre de modéliser une baie avionique en considérant des modèles réduits pour chaque équipement. Le multi-physique peut par exemple être utile pour quantifier les effets thermiques dans une modélisation électrique, les pertes par effet Joule en étant dépendantes.
2. **Optimisation / Propagation d'incertitudes**. Ces deux techniques peuvent s'avérer nécessaire afin de concevoir, ou dimensionner, des équipements électroniques. Elles peuvent en revanche être très gourmandes en nombre d'appels au modèle thermique, ce qui justifie le besoin de modèles réduits.

Modèles réduits en thermique électronique La réduction de modèle consiste à construire un modèle « simplifié » à partir de données plus détaillées. Cette simplification peut par exemple se traduire par la réduction du nombre de nœuds du maillage (dans le cas des modèles nodaux), dans la simplification du modèle physique, ou encore en ajustant des modèles analytiques à un jeu de données évalué sur un maillage fin. Le modèle réduit a donc beaucoup moins de sorties et de paramètres que le modèle de référence. Cette réduction est très souvent menée à partir d'un modèle détaillé, considéré trop gourmand en temps de calcul et que l'on souhaite remplacer par un modèle moins coûteux.

La méthode la plus classiquement utilisée en thermique est la réduction par modèle nodal. L'idée est de modéliser le comportement thermique par un système d'équations différentielles ordinaires (cf équation (2.7)), donc conservant la physique, mais en considérant un nombre d'éléments très restreint : il est par exemple fréquent de passer d'un million d'éléments dans un modèle détaillé, à moins d'une centaine d'éléments dans un modèle nodal réduit. Cette réduction se décompose en plusieurs étapes :

1. Choisir les conditions aux limites et termes forçants nominaux à considérer sur le modèle 3D. Ce sont les calculs à partir desquels le modèle réduit est construit. Par exemple, la température ambiante en croisière dans la baie avionique, la puissance à dissiper maximale, le débit d'extraction de l'air du ventilateur en marche...
2. Par l'expertise d'un thermicien, définir le nombre et la position des nœuds du modèle réduit. Il s'agit plus précisément de définir le nombre d'éléments N_{el} et la position des $\mathcal{V}_i$, $i \in \{1...N_{el}\}$ sachant que chacun de ces volumes sont supposés isothermes. Il ne faut donc pas réduire une zone à fort gradient de température en un seul volume.
3. Grâce aux résultats 3D lancés au point 1, calculer les flux de puissance conductifs, convectifs et radiatifs entre chacun de ces volumes : les ϕ_{ij} (définis par les équations (2.2) et (2.6)). Ceci permet de calculer les diverses conductances (équations (2.3) et (2.4)).
4. Tester le modèle en le comparant à un modèle 3D sur des cas lancés au point 1 et non utilisés au point 3. Si l'erreur est trop importante, retourner au point 2.

Ce que l'on remarque dans ce processus, c'est la part importante laissée à l'expertise de l'ingénieur. Cette caractéristique rend cette méthode difficilement automatisable.

Une autre limite non négligeable, et qui justifie le besoin d'autres méthodes de réduction proposées dans cette thèse, est la difficulté de la prise en compte par un unique modèle réduit des différents régimes de fonctionnement de l'équipement. Par exemple, les équipements étudiés sont fréquemment équipés d'un ventilateur afin d'évacuer la chaleur dissipée par les différents composants à l'extérieur de l'équipement. Lorsque ce ventilateur est en marche, la convection à l'intérieur est du type forcé (défini à la section 2.2.1.1). Or, si ce ventilateur tombe en panne, la convection devient naturelle. De plus, ces deux types de convection sont fondamentalement différents, la convection naturelle étant fortement non-linéaire. Ceci rend la modélisation de ces deux régimes par un unique modèle réduit difficile. Notamment, les techniques de réduction thermiques détaillées ici n'en sont pas capables : les erreurs commises par un modèle ajusté à partir d'un calcul 3D avec un ventilateur en marche peuvent être supérieures à 100% s'il est utilisé pour prédire le fonctionnement en cas de panne. Il est donc nécessaire de construire un modèle nodal réduit différent par régime, ce qui complexifie la modélisation du passage d'un fonctionnement nominal à un cas de panne.

Il serait donc intéressant de disposer de la construction d'un **unique** modèle réduit capable de prédire le fonctionnement nominal **et** le fonctionnement en cas de panne du ventilateur. Ce problème sera abordé au chapitre 9.

2.2.2 Modélisation de l'effet de la température sur la durée de vie

Pour pouvoir comparer l'équipement de référence et le nouvel équipement dans la nouvelle position, il faut définir une ou des mesures de performance. Toutefois, la thermique

ne permet pas d'améliorer le fonctionnement de l'équipement. En effet, même s'il existe des températures optimales de fonctionnement, il est illusoire d'espérer fixer la température d'un équipement autour d'une valeur précise. En revanche, une température trop élevée des composants électroniques peut les endommager, et donc nuire à leur fonctionnement à long terme. De plus, il suffit qu'un seul composant électronique ne fonctionne plus pour que l'équipement entier devienne inutilisable. La durée de fonctionnement de l'équipement correspond donc à la durée de fonctionnement de son composant électronique le plus critique. C'est par exemple celui dont le matériau est le plus fragile, ou celui qui a le plus d'énergie à dissiper. Afin d'estimer l'effet de la température sur la durée de vie du

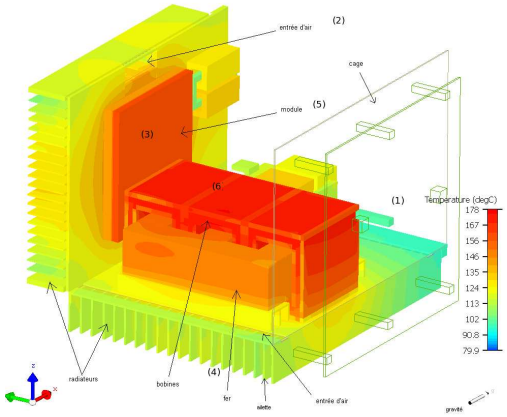


FIGURE 2.3 – Illustration des températures de composants électroniques à l'intérieur d'un équipement issues d'un calcul FLOTHERM

composant critique, il faut comprendre comment la variation de température peut dégrader la structure d'un matériau. Deux cas sont à considérer :

1. Chaque matériau a une température critique, T_{max} (voir schéma 2.4), au-delà de laquelle il risque de fondre ou d'être détruit. Dans le cas d'un composant électronique, le dépassement de ce T_{max} rend l'équipement inutilisable. Dès lors, ce cas de figure est inenvisageable et il doit être assuré par le calcul qu'il ne risque pas d'arriver.
2. Les variations de température imposent des contraintes sur la structure d'un matériau. De ce fait, si la température d'un matériau subit des variations de température en fonction du temps trop importantes, ces variations peuvent provoquer des déformations qui endommagent la structure. Ceci implique qu'au cours d'un vol, les composants sont sollicités en déformation de manière répétée, ce qui a pour conséquence leur fatigue et donc leur endommagement. À terme, cette fatigue est la cause de la destruction de l'équipement. Une mesure de performance, ou plutôt de non dégradation de performance, peut donc être proposée : **la durée de vie de l'équipement en fonction des variations de la température du composant critique au cours du temps.**

Calcul de la durée de vie Pour calculer la durée de vie de ce composant critique, il faut faire appel à la notion d'endommagement provenant de la structure. Elle consiste en l'apparition de cavités dans un volume de matière, induites par les efforts que subit le

matériau. L'accumulation de ces fissures est ce qui mène à la rupture, et donc à la fin de vie du composant étudié. Ces phénomènes sont traduits par la variable d'endommagement D, définie dans le livre [Lemaitre et Chaboche, 2004]. Elle correspond à la densité surfacique des discontinuités de la matière, **rendue sans dimension** grâce à la surface totale. Ce qui signifie que $D = 0$ correspond à un état non endommagé ou vierge, $0 \leqslant D < 1$ correspond à un état d'endommagement et enfin $D = 1$ traduit le moment où l'élément de volume se rompt en deux parties, donc l'état de rupture. En ce qui concerne les sources d'endommagement, il en existe deux principales :

- les endommagements dus à de grandes déformations plastiques ou à des déformations par fluage,
- les endommagement de fatigue : sous l'action de sollicitations répétées (ici les variations de températures que subit le composant), périodiques ou non, des microfissures se créent dans le matériau et provoquent à la longue la rupture. L'étude de l'endommagement de fatigue se fait en décomposant les efforts subits par la structure en différents cycles (périodiques ou non).

Dans cette étude, les seules déformations considérées sont celles induites par la température. Or les sauts de température au cours d'un vol n'ont pas une amplitude importante mais sont cycliquement répétés, étant la réponse du système à une succession de profils de vol types (cf figure 2.7 où les conditions aux limites sont constamment de ce type). Ce sont donc des sollicitations faibles mais répétées que subit l'équipement et ceci implique que l'endommagement dont il est question ici est uniquement un endommagement de fatigue.

Pour calculer le lien entre cette variable et la durée de vie, il faut faire plusieurs hypothèses :

1. Seuls les effets sur le composant critique sont regardés, étant le composant le plus fragile donc le plus limitant.
2. L'approximation est faite de n'utiliser qu'un unique profil de vol moyen pour la pression, la température et la puissance à dissiper (cf figure 2.7) à imposer au modèle d'équipement afin de calculer le profil de température du composant critique (cf section 2.3). Il s'agit donc de calculer le nombre de vols moyens N_{fvol} réalisables par l'équipement avant la rupture du composant critique.
3. Seules les contraintes dues aux sauts de température du composant électronique sont considérées pour calculer l'endommagement. Toutes les autres sources de contraintes et déformations sont donc négligées.
4. La loi du cumul linéaire des endommagements (ou loi de Miner) est utilisée. Elle est adaptée au cas où les chargements sont d'amplitude et de valeur moyenne peu variables. Elle permet de faire le lien entre l'endommagement total D et les endommagements provoqués par les sauts de température à chaque vol D_i (équation (2.8)).

Soit N_{jump} le nombre de sauts de température imposés au composant à chaque profil de vol. La condition 2 permet de définir un cycle périodique dans lequel chaque saut de température ΔT_i ($i \in \{1...N_{jump}\}$) que subit le composant critique (cf point 3 et voir figure 2.4) correspond à un endommagement D_i de ce dernier. Grâce au point 3, chacun de ces endommagements est uniquement caractérisée par le nombre N_{fi} de cycles avant rupture qu'impliquerait cette contrainte, c'est-à-dire par le nombre de cycle avant rupture si les cycles n'étaient composés que de cette contrainte. En quelque sorte, cela peut être vu comme la part d'endommagement du composant due à ce saut de température.

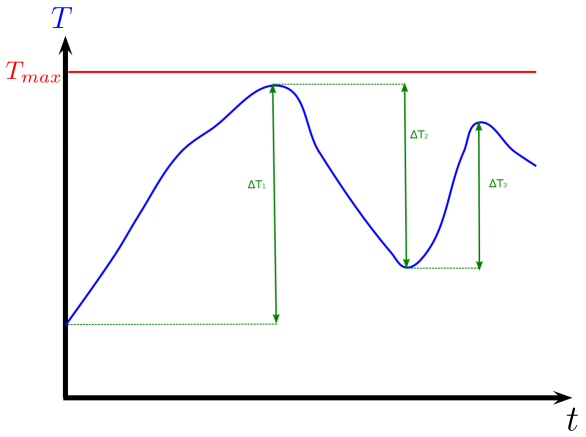


FIGURE 2.4 – Représentation de la température du composant critique avec le seuil T_{max} à ne pas dépasser et les sauts de température ΔT_i engendrant une fatigue du matériau de ce dernier

Soit N_{vols} le nombre de vols réalisables avant rupture, la loi du cumul linéaire du point 4 s'écrit donc :

$$D = \sum_{i=1}^{N_{jump}} D_i = \sum_{i=1}^{N_{jump}} \frac{N_{vols}}{N_{fi}} = N_{vols} \sum_{i=1}^{N_{jump}} \frac{1}{N_{fi}} \tag{2.8}$$

En prenant $D = 1$ dans l'équation (2.8) (définition de la rupture), la valeur de N_{fvol}, le nombre de vols réalisables avant rupture, peut être calculé :

$$N_{fvol} = \frac{1}{\sum_{i=1}^{N_{jump}} \frac{1}{N_{fi}}} \tag{2.9}$$

Enfin, le guide FIDES [Miller *et al.*, 2010] fournit la loi de Coffin-Manson [Lemaitre et Chaboche, 2004] permettant de calculer le lien entre le nombre de cycles avant rupture et les contraintes imposées au composant. Grâce à l'hypothèse 3 et en posant C_{fides} une constante dépendant des propriétés du matériau (la même pour toutes les sauts de température), le nombre de cycles avant rupture se calcule grâce à l'équation (2.10).

$$\frac{1}{N_{fi}} = C_{fides} e^{-1/T_{maxcycle_i}} (\Delta T_i)^{1.9} \tag{2.10}$$

La mesure de performance recherchée est dès lors obtenue en réinjectant le résultat de 2.10 dans l'équation (2.9). Elle représente (équation (2.11)) donc le nombre de vols possibles avant rupture en ne prenant en compte que les effets thermomécaniques sur le composant.

$$\boxed{N_{fvol} = \frac{1}{C_{fides} \sum_{i=1}^{N_{jump}} e^{-1/T_{maxcycle_i}} (\Delta T_i)^{1.9}}} \tag{2.11}$$

Remarque sur la reproductibilité du choix des cycles ΔT_i à partir d'un profil de température :
Le guide FIDES [Miller *et al.*, 2010] fournit des règles qui doivent être appliquées pour une bonne représentativité et une bonne reproductibilité de la description des cycles thermiques. Ces règles stipulent notamment qu'un cycle thermique doit correspondre à un phénomène identifié générant la contrainte. Par exemple, une mise sous tension, la montée en altitude, une surchauffe liée à l'état du système ... Dans le cas de l'équipement étudié ici et défini à la section 2.3, le profil de vol (figure 2.7) identifie clairement chacune de ces phases, ce qui permet aisément de choisir les ΔT_i d'intérêt.

2.2.3 Modélisation des incertitudes - Introduction de mesure de robustesse

Dans cette partie il est uniquement question des incertitudes sur les entrées du modèle, celles dues au modèle lui-même étant hors du cadre de cette étude. Dans les sections précédentes, les équations et leurs conditions aux limites ont été détaillées. Par exemple ces dernières sont définies grâce aux profils de vol définissant les variables environnementales ou encore grâce aux technologies de refroidissement utilisées. Toutes ces variables ne sont toutefois pas connues avec précision du fait du positionnement amont de cette étude. Soit f_1 la fonction de durée de vie, soit $\boldsymbol{x}$ les variables de contrôle de l'optimisation et soit $\boldsymbol{\xi}$ les incertitudes entachant les différents paramètres, aussi bien de contrôle qu'environnementaux.

Sachant que les paramètres incertains sont nécessaires pour calculer la durée de vie, cette dernière est également incertaine. Si l'on se place dans un cadre probabiliste, ceci signifie que la durée de vie est une variable aléatoire au même titre que les entrées permettant de l'estimer. Or, il n'y a aucun sens à vouloir minimiser une variable aléatoire. Pour appliquer un algorithme d'optimisation, il est nécessaire de modifier l'objectif incertain à l'aide d'une mesure de robustesse. C'est une mesure qui associe une valeur réelle à une variable aléatoire. Selon la mesure de robustesse utilisée cette valeur réelle quantifie différentes caractéristiques de la variable aléatoire étudiée. De sorte à choisir la mesure de robustesse la plus adaptée, il faut que l'architecte sache précisément quel est l'effet des incertitudes qu'il souhaite quantifier sur son objectif. En d'autres termes, veut-il lisser les effets des incertitudes en s'intéressant uniquement à la moyenne de l'objectif quelles que soient les incertitudes, ou souhaite-t-il s'assurer qu'il en estime sa pire valeur possible. Dans le cas de la durée de vie d'un équipement électronique, l'architecte ne veut en aucun cas la surestimer car une surestimation peut avoir de graves conséquences. En effet, il est inacceptable de se dire que la valeur estimée peut ne pas être la pire, d'autant plus en avant projet où l'on souhaite avoir des valeurs dimensionnantes. Il ne faut donc pas s'intéresser aux mesures quantifiant les effets moyens mais plutôt les mesures quantifiant la queue d'une distribution.

Ces mesures sont de deux types :

— Déterministes, c'est-à-dire que les incertitudes sont uniquement définies par l'intervalle dans lequel elles peuvent prendre leur valeur, par exemple $\xi \in [a, b]$. La seule mesure possible dans ce cas-là est le pire cas $\max_{\xi}(f_1(x, \xi))$. En revanche, du fait que les lois des incertitudes en entrée ne sont pas prises en compte, c'est-à-dire la probabilité d'occurrence de chacune des valeurs possibles, le voisinage du point x où tombe ce maximum peut avoir une probabilité d'occurrence très faible. Du

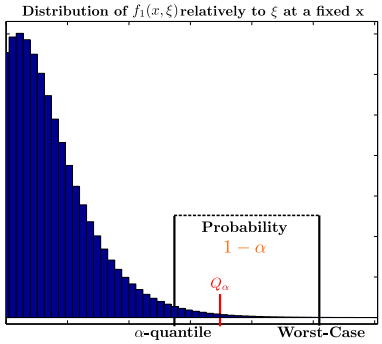


FIGURE 2.5 – Illustration des mesures de robustesse estimant les queues de distribution

fait que la zone où se trouve le maximum a une petite probabilité d'occurrence, cette mesure peut être trop conservative. D'où l'intérêt des mesures probabilistes qui permettent de pénaliser les zones de faibles probabilités.

— Probabilistes, c'est-à-dire que la forme des incertitudes en entrée a son importance :
  — α-quantile $q_\alpha(f_1)$ avec $q_\alpha(f_1)$ tel que $Pr_\xi(f_1(x,\xi) \leqslant q_\alpha(f_1)) = \alpha$. Toutefois, cette mesure a une limite : elle ne donne aucune information sur les évènements qu'elle ne quantifie pas, c'est-à-dire les $(1-\alpha)\%$ des cas les plus extrêmes.
  — Une mesure permettant de pondérer l'effet de ces évènements est l' α-superquantile Q_α. Elle est définie de la manière suivante :

$$Q_\alpha(f_1) = \frac{1}{1-\alpha}\int_\alpha^1 q_u(f_1)\ du$$

Elle est un élargissement du quantile puisqu'elle pondère tous les évènements situés entre ce dernier et le pire cas. Elle est de ce fait sensible à la forme de la queue de distribution, à la différence du quantile.

2.3 Équipement électronique étudié

L'équipement étudié dans le cadre du projet européen TOICA [1] (*Thermal Overall Integrated Conception of Aircraft*) est un TRU (pour *Transformer-Rectifier Unit* voir figure 2.6). C'est un transformateur électrique permettant de modifier la tension et le courant provenant d'une source d'énergie électrique alternative, tout en conservant la même fréquence et la même forme. Or, cette transformation du courant peut dissiper une quantité de puissance importante. En outre, plus la puissance à dissiper est grande, plus les composants électroniques chauffent. Voilà pourquoi cet équipement est un bon exemple pour mettre en place des méthodes visant à minimiser la perte de durée de vie due à la surchauffe des composants électroniques.

L'équipement de référence, celui provenant d'un avion existant, est une boîte en aluminium contenant divers composants électriques et électroniques. Ce sont eux qui dissipent de la puissance et donc chauffent le système. Par exemple, on peut citer :

— Des bobinages, dissipant de la chaleur mais peu fragile
— Divers composants électroniques (par exemple des transistors)

1. http ://www.toica-fp7.eu/

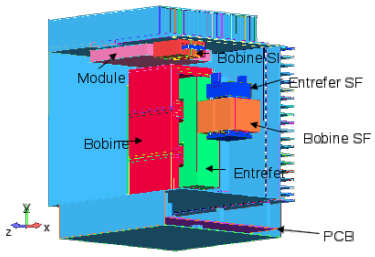


FIGURE 2.6 – Schéma en coupe d'un TRU

Pour garder la température maximale dans l'équipement en dessous des seuils d'endommagement, il est nécessaire de mettre en place des systèmes de refroidissement permettant d'évacuer une certaine quantité d'énergie. Dans l'équipement de référence, il y a uniquement un ventilateur faisant circuler l'air depuis l'extérieur vers l'intérieur, et donc extraire par convection cette chaleur. Enfin, des radiateurs peuvent être fixés aux composants critiques, permettant d'augmenter les échanges par convection avec l'air traversant l'intérieur du boîtier.

Cet équipement n'est en fonctionnement qu'au cours des vols effectués par l'avion. Les profils de température, de pression ou encore de puissance électrique imposée se ressemblent fortement d'un vol à un autre. Dans le cadre de cette étude, une approximation est faite en utilisant un profil de vol "type", c'est-à-dire détaillant un vol décomposé en 4 étapes : la première étape correspond à un avion garé sur la piste (taxi) dans une région au climat tempéré (température extérieure à 15 °C). La température dans la cavité étudiée augmente à cause du rayonnement solaire et la puissance est nulle étant donné que tout est éteint. La deuxième étape est celle de l'allumage de l'avion et du décollage. La troisième étape décrit la phase de croisière où les constantes physiques se stabilisent par rapport au décollage. Enfin, la dernière phase est la phase d'atterrissage. Dans l'exemple de la figure 2.7, l'avion atterrit dans une région au climat tempéré. Ces profils correspondent à des conditions aux limites ou à des termes forçants des modèles thermiques utilisés.

2.3.1 Solutions de refroidissement et définition de l'objectif de masse et de coût

Les technologies de refroidissement interviennent à différentes échelles. Cela peut aller du composant jusqu'à la baie avionique entière. Ici, il n'est question que des technologies adaptées à l'équipement. Le principe d'une technologie de refroidissement est d'évacuer la chaleur dissipée par ce dernier dans l'environnement dans lequel il se trouve, en augmentant les échanges thermiques entre les deux.

Les solutions de refroidissement peuvent évacuer la chaleur par les trois types d'échanges de flux thermique définis dans la section 2.2.1.1. Voici une liste des solutions de refroidissement envisageables en aéronautique :

— Par rayonnement des parois externes vers une source froide : une solution de refroidissement pesant très peu mais pouvant être très coûteuse est le traitement de la surface de l'équipement de sorte à augmenter les échanges radiatifs. Ceci permet d'augmenter l'émissivité ε des parois de l'équipement et donc d'évacuer plus d'énergie par rayonnement. Dans les équations, elle est uniquement présente dans le calcul du flux radiatif à l'équation (2.6) puisqu'elle ne joue que sur cette émissivité ε.

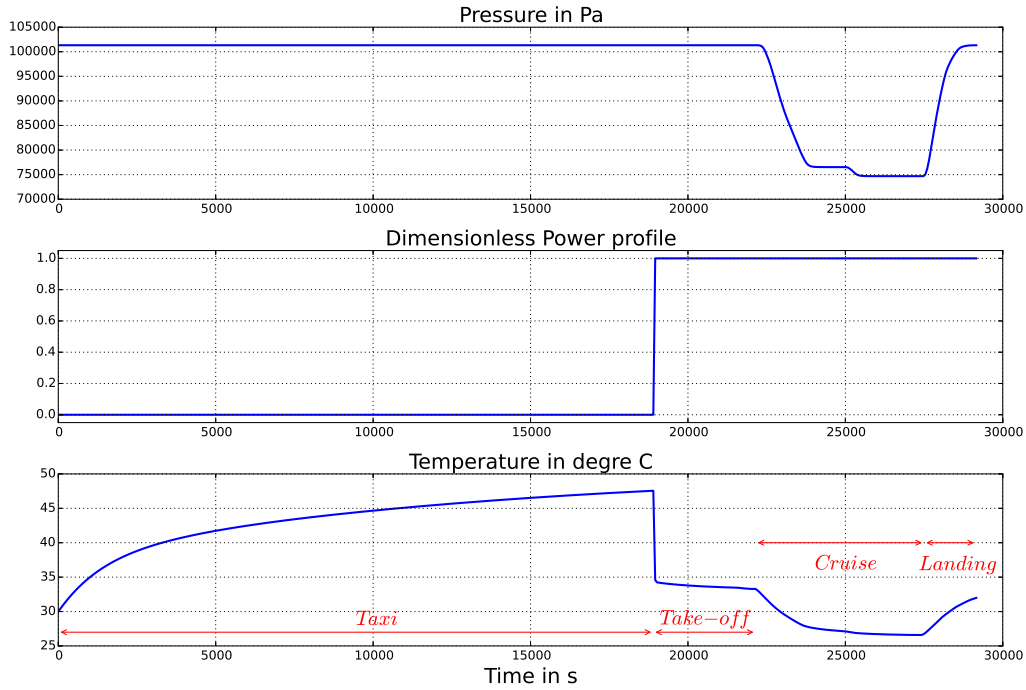


FIGURE 2.7 – Profil de vol type avec stationnement en zone tempérée et atterrissage en zone tempérée

— Par convection : cela peut se traduire par l'utilisation d'un ventilateur ou d'une pompe permettant d'extraire la chaleur de l'équipement par convection interne. Cette extraction d'air est dimensionnée en fonction de la puissance à dissiper par l'équipement. Le standard ARINC impose une différence de 15 °C entre l'entrée et la sortie d'air de l'équipement en fonctionnement nominal. Une des limites de cette technique est la possibilité de panne d'un ventilateur, ces derniers ayant une fiabilité modérée. Au niveau de la modélisation thermique, la modification de la ventilation n'intervient que dans la modification du débit massique d'extraction de l'air $\dot{m}$ permettant de calculer les échanges par convection.

— Par extraction de la puissance par contact avec les parois externes de l'équipement : cela peut être traduit par une résistance thermique $R_{th} = \frac{1}{C_{th}}$ (C_{th} défini à l'équation (2.3)) équivalente appliquée sur une paroi externe à laquelle on peut associer une source froide qui peut être différente de la température ambiante. On peut par exemple citer :

1. la pose de radiateurs sur les parois externes de l'équipement est une première solution envisageable. Elle permet d'augmenter la surface d'échange convectif avec l'environnement extérieur (le S_{ij} dans l'équation (2.4)) et donc d'évacuer une quantité de chaleur plus importante par convection. Actuellement très utilisée en aéronautique, elle présente l'avantage de peser peu et de coûter peu. Son principal inconvénient réside dans le peu d'efficacité qu'elle peut avoir, surtout si l'équipement se trouve dans une zone mal ventilée.

2. une solution largement moins présente en aéronautique du fait de son coût potentiellement rédhibitoire : l'usage de boucles diphasiques afin d'extraire la

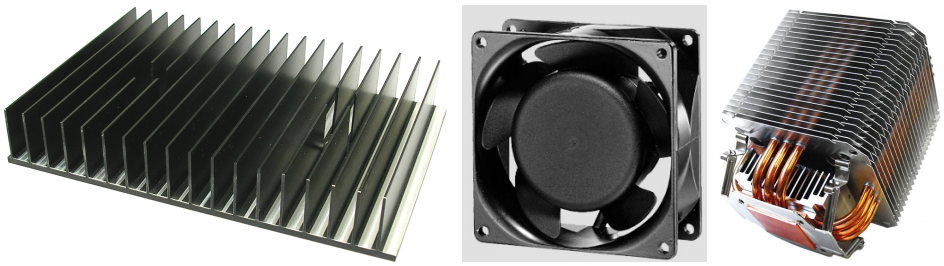


FIGURE 2.8 – Illustration d'un radiateur, d'un ventilateur et de caloducs (tubes en cuivre en bas de l'image)

chaleur des parois de l'équipement vers une source froide. C'est en fait un tube rempli par un fluide à saturation permettant de transporter la chaleur du point chaud (la paroi) vers le point froid (l'environnement). Son principe est le suivant : le point chaud permet l'évaporation d'un liquide qui remonte ainsi jusqu'au point froid (l'environnement) où il se condense et retourne à l'état liquide. L'avantage de cette méthode est la conductivité équivalente qu'elle permet d'obtenir : environ 100 fois plus que le métal le plus conducteur si elle est utilisée dans de bonnes conditions.

3. une dernière solution envisageable pourrait être le *spray cooling*. C'est un système permettant d'envoyer à haute pression un fluide sur l'élément à refroidir. Ce dispositif est lourd et coûteux. Il est donc difficilement envisageable en aéronautique où ces deux paramètres sont primordiaux.

2.3.2 Choix des variables de contrôle du problème d'optimisation

Afin de traduire le problème industriel en problème d'optimisation, il est nécessaire d'identifier les variables de contrôle de l'optimisation $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$, c'est-à-dire les paramètres sur lesquels l'architecte peut jouer pour minimiser les objectifs. Dans cette étude, ils correspondent à des traductions mathématiques des solutions de refroidissement envisageables. Chacun de ces paramètres fait varier l'objectif de masse et de coût et permet de définir l'un des objectifs à minimiser. Les fonctions de masse et de coût liées à chaque paramètres sont définies en figures 2.10, 2.12, 2.9, et 2.11. Elles sont exprimées en pourcentage de ceux de l'équipement de référence.

Ces paramètres de contrôle sont :

— L'émissivité des parois de l'équipement ε qui traduit l'effet d'un traitement de surface $x_1 := \varepsilon$ afin d'augmenter les échanges radiatifs avec l'environnement. Ce paramètre est sans dimension et peut varier entre $x_1 \in [0.1, 0.9]$. Il n'influe pas du tout sur la masse de l'équipement alors que son coût peut devenir très grand pour approcher des émissivités proches de 0.9. Ceci est représenté en figure 2.9.

— La résistance thermique équivalente R_{th} appliquée à la face sur laquelle se trouve le composant critique, $x_2 := R_{th}$. Ce paramètre permet de traduire différentes solutions techniques tout en utilisant un paramètre continu variant dans le domaine suivant $R_{th} \in [0.01, 10]$ K/W. Cet intervalle de variations est décomposé en sous-intervalles, chacun d'eux correspondant à une technologie de refroidissement différente, comme représenté en figure 2.10. Ces technologies vont de simples radiateurs utilisés afin d'augmenter les échanges convectifs (dans la zone $R_{th} \in [8, 10]$

K/W), en passant par des caloducs dont la qualité et le nombre varie (dans la zone $R_{th} \in [0.1, 8]$ K/W) jusqu'au *spray cooling* qui est une méthode très lourde et coûteuse pour la gamme $R_{th} \in [0.01, 0.1]$ K/W. La bijection choisie entre l'intervalle de R_{th} et les solutions de refroidissement potentielles permet uniquement de définir les fonctions de masse et de coût. Elles ne changent rien au niveau de la modélisation et c'est justement la raison pour laquelle il a été choisi de formuler une solution de refroidissement par une résistance thermique équivalente.

— La surface d'échange avec l'extérieur S_{wet} des cinq autres faces de l'équipement qui traduit la pose éventuelle d'ailettes sur ces faces $x_3 := S_{wet}$. C'est un paramètre sans dimension : ce n'est qu'un paramètre multiplicatif ajouté dans l'équation (2.4) traduisant les échanges convectifs entre les parois de l'équipement et l'air extérieur. Il varie donc dans l'intervalle $S_{wet} \in [0.9, 1.2]$. Le cas où $S_{wet} < 1$ peut correspondre au cas nominal car il traduit des pertes dans les échanges thermiques avec l'air extérieur par exemple à cause de la présence de câbles ou d'autres irrégularités non représentées. C'est la raison pour laquelle il n'entraîne pas de changement dans la masse et le coût de l'équipement sur la figure 2.11. Le cas où $S_{wet} > 1$ représente le cas où l'on place des ailettes de hauteur variable sur ces faces. Plus la hauteur d'ailette est grande, plus la masse et le coût sont importants, cette variation étant considérée linéaire.

— Le débit massique d'extraction de l'air $\dot{m}$ qui traduit la taille du ventilateur utilisé, $x_4 := \dot{m}$. Afin de pouvoir augmenter ce débit, il est nécessaire de changer de ventilateur pour passer d'une gamme de débit à une autre. Chaque changement induit donc un saut de masse et de coût comme illustré en figure 2.12. Au niveau de la variation de ce paramètre, on se donne la possibilité d'utiliser un ventilateur de taille plus réduite que celle de l'équipement de référence. Ce dernier a pour débit 0.025 kg/s, ce qui correspond à la valeur trouvée pour satisfaire la norme ARINC. On se donne donc la possibilité de trouver un $\dot{m}$ allant d'une convection presque naturelle jusqu'à deux fois la norme ARINC $\dot{m} \in [0.0035, 0.05]$ kg/s.

Du fait que la masse et le coût de ces solutions sont fortement corrélés, il est donc possible de les traduire en un seul objectif, appelé ici f_2. Ce dernier ne dépend pas des incertitudes puisque pour chaque solution technique choisie, il n'y a pas de doutes quant au prix ou au poids qu'elles nécessitent.

2.3.3 Incertitudes sur les paramètres

Sachant que cette étude est réalisée en phase d'avant-projet, les paramètres sont donc entachés d'une incertitude que l'on note $\boldsymbol{\xi}$. Ces incertitudes portent aussi bien sur les quatre variables de design $\boldsymbol{\xi}_x$ que sur des paramètres permettant de définir les conditions aux limites des modélisations thermiques à mettre en place, aussi appelés paramètre environnementaux $\boldsymbol{\xi}_e$. Voici une liste des incertitudes $\boldsymbol{\xi} = (\boldsymbol{\xi}_x, \boldsymbol{\xi}_e)$ à prendre en compte dans la formulation du problème d'optimisation.

Remarque : tous ces paramètres étant bornés, ces paramètres sont générés par des lois uniformes ou des lois normales tronquées. Ces lois de probabilités sont définies à l'annexe A. Dans le second cas, il faut donc que l'utilisateur fournisse les bornes des incertitudes M en plus de leur écart type σ.

— Sur les paramètres de design, les incertitudes sont des variables aléatoires suivant une loi normale tronquée $\mathcal{N}_{[-M,M]}(0, \sigma_{xi})$ de moyenne nulle et **indépen-**

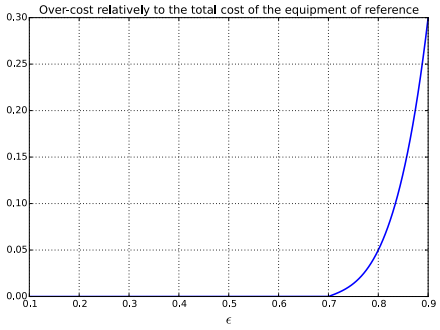


FIGURE 2.9 – Estimation du coût induit par le traitement de surface exprimé en pourcentage du coût de l'équipement de référence

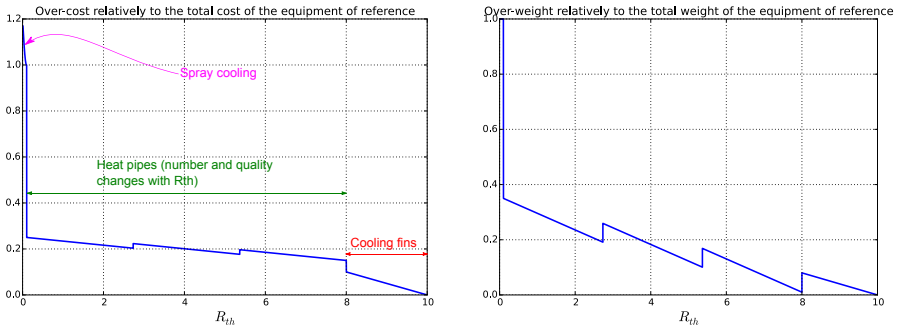


FIGURE 2.10 – Estimation du coût et de la masse induits par la modification de l'extraction par conduction en une paroi (exprimés en pourcentage des valeurs de l'équipement de référence)

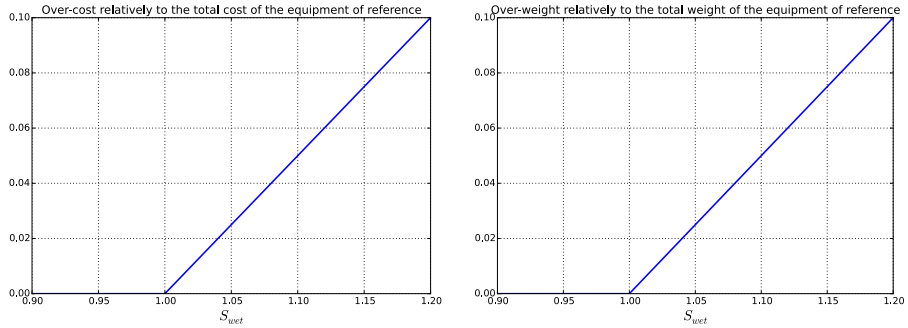


FIGURE 2.11 – Estimation du coût et de la masse induits par la modification de la surface mouillée des autres parois (exprimés en pourcentage du coût de l'équipement de référence)

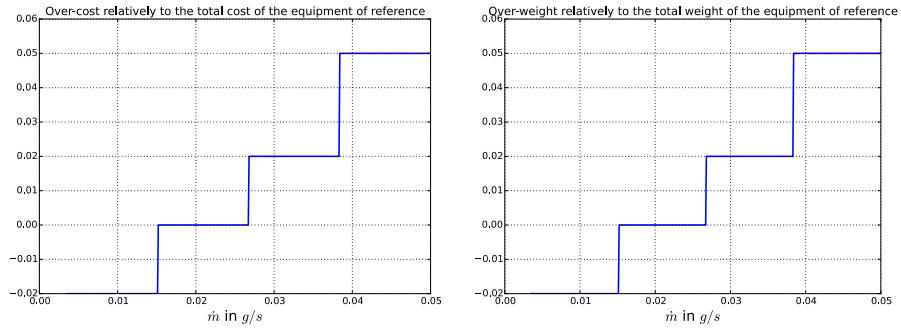


FIGURE 2.12 – Estimation du coût et de la masse induits par la modification du ventilateur (exprimés en pourcentage des valeurs de l'équipement de référence)

Paramètre	Type d'incertitudes	écart type σ	Borne M
$x_1 := \varepsilon$	Absolu	0.05	0.1
$x_2 := R_{th}$	Relatif	5%	10%
$x_3 := S_{wet}$	Relatif	1%	5%
$x_4 := \dot{m}$	Relatif	10%	20%

TABLEAU 2.1 – Paramètres des lois normales des incertitudes $\boldsymbol{\xi}_x$ sur les variables d'optimisation $\boldsymbol{x}$

dantes deux à deux. Elles sont des perturbations autour de l'état nominal $\boldsymbol{x}$. Selon la variable concernée, elles sont soit additives, $X_i = x_i + \xi_{x,i}$, soit relatives, $X_i = x_i(1 + \xi_{x,i})$. Les ordres de grandeur et le type des incertitudes sont données dans le tableau 2.1.

— Les paramètres environnementaux $\boldsymbol{e}(t)$ ont également des incertitudes de moyenne nulle et indépendantes deux à deux. Cependant, bien que ces paramètres dépendent du temps, leurs incertitudes ne sont que des perturbations indépendantes du temps $\boldsymbol{\xi}_e \sim \mathcal{N}_{[-M_{e,i},M_{e,i}]}(0, \sigma_{e,i})$. Les paramètres environnementaux incertains sont les suivants :

1. La température ambiante $e_1(t) := T_{conv}(t)$ est connue à une constante près $E_1(t) = e_1(t)+\xi_{e,1}$. L'incertitude sur ce paramètre n'étant qu'un biais de mesure, elle est modélisée par une loi uniforme sur un intervalle $\xi_{e,1} \in [-1, 1]$ °C.
2. Le coefficient d'échange convectif entre les parois externes de l'équipement et l'air environnant : $e_4(t) := h(t)$. De la même manière que pour la température, ces coefficients sont connus à une constante près : $E_2(t) = e_2(t) + \xi_{e,2}$. Cette incertitude suit une loi normale tronquée, donc de la forme suivante : $\xi_{e,2} \sim \mathcal{N}_{[-M_{e,2},M_{e,2}]}(0, \sigma_{e,2})$.
3. La puissance à dissiper par les composants électroniques : $e_3(t) := \Phi(t)$. Cette fois-ci, l'incertitude étant nulle lorsque cette puissance est nulle, et grande lorsque cette puissance est grande, cette incertitude est connue de manière relative : $E_3(t) = e_3(t)(1 + \xi_{e,3})$. Elle aussi suit une loi normale centrée en zéro. La seule différence ici est que l'écart type $\sigma_{e,3}$ ainsi que le $M_{e,3}$ (la borne pour la troncature de la loi) sont donnés en pourcentage.

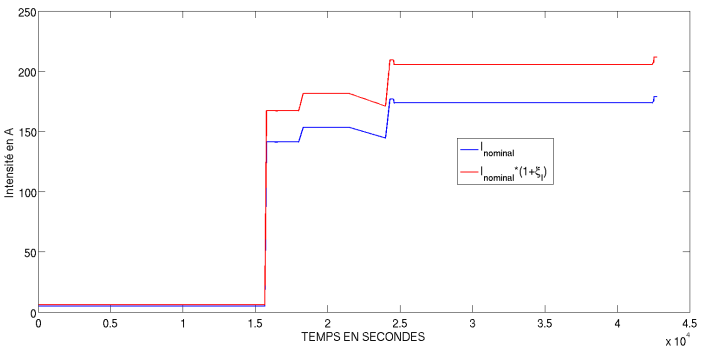


FIGURE 2.13 – Illustration de l'effet de la génération d'une incertitude relative sur un paramètre environnemental dépendant du temps

Paramètre	Loi	Type d'incertitudes	écart type σ	Borne M
$e_1(t) := T_{conv}(t)$	Uniforme	Absolu	Non Applicable	1°C
$e_2(t) := h(t)$	Normale	Absolu	1.0 W.K^{-1}.m^{-2}	2.0 W.K^{-1}.m^{-2}
$e_3(t) := \Phi(t)$	Normale	Relatif	10%	20%
$e_4(t) := P(t)$	Normale	Relatif	1%	5%

TABLEAU 2.2 – Paramètres des lois des incertitudes $\boldsymbol{\xi}_e$ portant sur les variables environnementales $\boldsymbol{u}(t)$

4. La pression ambiante $e_2(t) := P(t)$ est également connue de manière relative $E_4(t) = e_4(t)\,(1 + \xi_{e,4})$. L'incertitude suit également une loi normale centrée en zéro, $\xi_{e,4} \sim \mathcal{N}_{[-M_{e,4}, M_{e,4}]}(0, \sigma_{e,4})$.

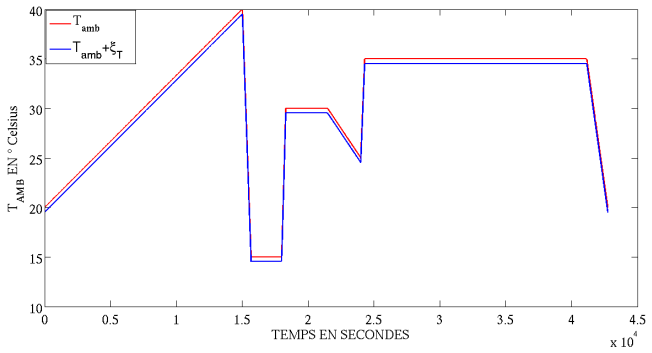


FIGURE 2.14 – Illustration de l'effet de la génération d'une incertitude additive sur un paramètre environnemental dépendant du temps

Les ordres de grandeur et la famille de lois de chacune des incertitudes environnementales sont résumés dans le tableau 2.2.

2.4 Optimisation sous incertitudes multi-objectif

Pour résumer, le problème industriel peut se ramener à la résolution du problème d'optimisation suivant :

$$\begin{cases} \min\limits_{x} \rho_\xi \left[f_1(\boldsymbol{x}, \boldsymbol{\xi}_x, \boldsymbol{\xi}_e) \right] \\ \min\limits_{x} f_2(\boldsymbol{x}) \\ \\ \underline{\text{sous contraintes :}} \\ \max\limits_{t,\xi} T(\boldsymbol{x}, \boldsymbol{\xi}_x, \boldsymbol{\xi}_e, t) < T_{max} \end{cases} \tag{2.12}$$

$$\boldsymbol{x} = (\varepsilon, R_{th}, S_{wet}, \dot{m}) \in \mathcal{D}_x \subset \mathbb{R}^4$$

$\mathcal{D}_x$ est le compact de $\mathbb{R}^4$ définissant les intervalles de variation, donnés en section 2.3.2, des variables d'optimisation. Ces variables correspondent à l'émissivité radiative des parois de l'équipement ε, la résistance thermique équivalente R_{th} appliquée à la paroi du composant critique, la surface mouillée d'échange avec l'air environnant S_{wet} et le débit massique d'extraction de l'air de l'équipement $\dot{m}$. Les incertitudes $\boldsymbol{\xi}$ sont définies en section 2.3.3. Elles entachent les quatre variables d'optimisation ainsi que quatre paramètres

environnementaux tels que la température ambiante et la pression ambiante de la zone où se trouve l'équipement, les coefficients d'échange convectif des parois externes ou encore la puissance à dissiper par les composants de l'équipement. ρ_ξ peut être une mesure de robustesse déterministe dans le cas d'un pire cas ou probabiliste dans le cas du superquantile. $T(\boldsymbol{x}, \boldsymbol{\xi}_x, \boldsymbol{\xi}_e, t)$ est la température du composant critique calculée avec des conditions aux limites et des termes forçants définissant un profil de vol type défini en figure 2.7. Pour rappel, f_2 correspond à la fonction de masse et de coût des solutions de refroidissement associées à un paramètre de design $\boldsymbol{x}$. Cette fonction est donnée par la somme des fonctions des figures 2.9, 2.10, 2.11 et 2.12. Elle ne dépend pas des incertitudes puisque pour chaque solution technique choisie, il n'y a pas de doutes quant au prix ou au poids qu'elles nécessitent.

La contrainte sur la température maximum du composant critique est nécessaire afin de garantir que ce composant ne soit pas dégradé au cours des profils de vol étudiés. En pratique, cette température maximum varie dans le sens contraire au maximum de la durée de vie quelles que soient les incertitudes en chaque $\boldsymbol{x}$. Ceci signifie que l'on ne peut pas rater de solutions en ne prenant pas en compte la contrainte : en effet, à cause de cette caractéristique, le domaine des solutions admissibles est délimité par une droite représentant une isovaleur de durée de vie. On peut donc calculer le front entier sans se soucier de cette contrainte, puis vérifier les solutions non admissibles *a posteriori* sans manquer de solutions dans l'espace des points admissibles.

Deuxième partie

État de l'art

Chapitre 3

Présentation des techniques d'apprentissage statistique pour la construction de modèles de substitution

Sommaire

Les simulations numériques permettant de modéliser un phénomène physique complexe ont souvent des temps d'exécution de plusieurs minutes voire plusieurs heures. Or, les techniques d'optimisation ou de propagation d'incertitudes peuvent nécessiter de nombreuses évaluations. Il est donc primordial de développer des techniques de réduction de modèle requérant peu d'appels aux simulations numériques de référence. Ces modèles numériques, provenant souvent de codes de calcul industriels, ne peuvent fournir rien d'autre à l'utilisateur que les sorties à des entrées données. On appelle cela des modèles entrées-sorties ou boîte noire, c'est-à-dire :

$$\begin{array}{rcccl} f & : & \mathcal{D}_x \subset \mathbb{R}^{N_x} & \longrightarrow & \mathbb{R}^{N_y} \\ & & \boldsymbol{x} & \longmapsto & f(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_{N_y}(\boldsymbol{x})) \end{array}$$

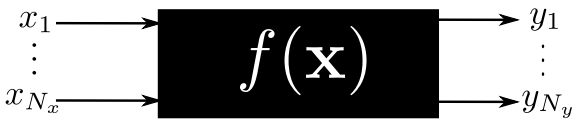


FIGURE 3.1 – Illustration d'un modèle entrée-sortie de type boîte noire

Même s'il existe des codes de calcul industriels dont la dérivée de la sortie peut être connue, par exemple par différentiation automatique, le plus souvent la dérivée de la sortie par rapport aux entrées n'est pas connue.

L'objectif de cette partie est donc d'expliquer les méthodes existantes dans le domaine de l'apprentissage statistique pour la construction de modèles de substitution de simulations numériques de type boîte noire, potentiellement coûteuses. L'apprentissage statistique n'étant par définition pas intrusif, c'est-à-dire qu'il ne nécessite pas de connaître les mécanismes permettant de passer des entrées aux sorties, son utilisation est tout à fait appropriée ici. Son principe est d'abord détaillé. Puis, les différents types de modèles de substitution couramment utilisés sont listés et expliqués. On s'intéresse plus particulièrement aux modèles de substitution de type krigeage, dont l'utilisation est très répandue en optimisation, et aux réseaux de neurones artificiels, dont l'application au cas spatio-temporel a déjà fait ses preuves.

Finalement, la dernière partie présente l'adaptation des techniques d'apprentissage statistique à la construction de modèles de substitution spatio-temporels.

3.1 Apprentissage Statistique

L'apprentissage statistique ne sert pas uniquement à faire de la régression. En effet, [Hastie *et al.*, 2009], [Vapnik, 1998] détaillent la théorie complète en expliquant également son aptitude à la classification. Appliquée à la réduction de modèles, elle consiste à approcher un modèle $f(\boldsymbol{x})$ uniquement à partir de simulations provenant de ce dernier. Ces simulations constituent l'ensemble d'apprentissage $DOE = \{(\boldsymbol{X}, \boldsymbol{Y})\}$ avec $\boldsymbol{X} = (\boldsymbol{x}^i)_{i\in[1,N_{DOE}]}$ et $\boldsymbol{Y} = (\boldsymbol{y}^i = f(\boldsymbol{x}^i))_{i\in[1,N_{DOE}]}$ aussi appelé plan d'expériences (ou *Design Of Experiment* (DOE) en anglais). Afin de générer ce DOE, il est nécessaire de choisir l'ensemble des entrées $\left\{(\boldsymbol{x}^i)_{i\in[1,N_{DOE}]}\right\}$ sur lesquels le modèle à approcher doit être estimé. Le choix du DOE générant les entrées est détaillé à la section 3.1.3. Une fois que ce DOE est généré, il permet par la suite de construire le modèle de substitution, qui est un modèle analytique à temps de réponse rapide $\hat{f}(x;\ \boldsymbol{w})$, paramétré par des variables $\boldsymbol{w} \in \mathbb{R}^{N_w}$, qui permettent d'ajuster le modèle aux sorties.

$$\begin{array}{rccl} \hat{f}\left(\bullet;\ \boldsymbol{w}\right) & : & \mathcal{D}_x \subset \mathbb{R}^{N_x} & \longrightarrow & \mathbb{R}^{N_y} \\ & & \boldsymbol{x} & \longmapsto & \hat{f}\left(\boldsymbol{x};\ \boldsymbol{w}\right) = \left(\hat{f}_1\left(\boldsymbol{x};\ \boldsymbol{w}\right), \cdots, \hat{f}_{N_y}\left(\boldsymbol{x};\ \boldsymbol{w}\right)\right) \end{array}$$

Ce type de modèle est communément appelé modèle de substitution, métamodèle ou encore surface de réponse, cette dernière appellation provenant plutôt du domaine de l'optimisation. En ce qui concerne les variables $\boldsymbol{w}$, elles dépendent de la forme analytique choisie, c'est-à-dire du type de modèle de substitution choisi. La dimension de ces paramètres,

$N_{\boldsymbol{w}}$, dépend de la complexité du modèle $C \in \mathbb{N}$:

$$N_{\boldsymbol{w}} = \mathcal{C}\left(C, N_x, N_y\right) \text{ avec } \begin{array}{rccc} \mathcal{C} & : & \mathbb{N}\times\mathbb{N}\times\mathbb{N} & \longrightarrow & \mathbb{N} \\ & & C, N_x, N_y & \longmapsto & \mathcal{C}\left(C, N_x, N_y\right) \end{array} \tag{3.1}$$

Cette complexité se traduit différemment selon la famille de fonction paramétrique choisie. Par exemple dans le cas d'une régression polynomiale, elle correspond au degré du polynôme. Le choix de cette complexité est l'une des difficulté de l'apprentissage statistique. Ce problème est d'ailleurs traité en section 3.1.2.

En ce qui concerne les modèles de substitution, on peut notamment les classer en deux types :

— Interpolants. Ce qui signifie que le modèle de substitution est exact aux points du DOE fourni, donc $\hat{f}(\boldsymbol{x}^i;\ \boldsymbol{w}) = f(\boldsymbol{x}^i)$ si $(\boldsymbol{x}^i, f(\boldsymbol{x}^i)) \in DOE$.

— Régressants. Ce sont les modèles non-interpolants donc commettant une erreur sur les points du DOE.

Afin d'ajuster le modèle de substitution au modèle de référence f à partir des points du DOE, il est nécessaire, quelle que soit la forme du modèle de substitution choisi, de rechercher les paramètres $\boldsymbol{w}$ optimaux solutions du problème d'optimisation de l'équation (3.2).

$$\boldsymbol{w} = \underset{v\in\mathbb{R}^{N_w}}{\operatorname{argmin}}\ \mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{v}\right);\ DOE) \tag{3.2}$$

$\mathcal{E}$ correspond à la fonction à minimiser pour construire le modèle $\hat{f}\left(\bullet;\ \boldsymbol{v}\right)$ à partir du plan d'expériences DOE. Si le modèle est régressant, on peut par exemple minimiser l'erreur quadratique moyenne (RMSE) commise sur le DOE :

$$\mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE) = \frac{1}{N_{DOE}} \sum_{\boldsymbol{x}^i\in DOE} \left\| \hat{f}(\boldsymbol{x}^i;\ \boldsymbol{w}) - f(\boldsymbol{x}^i) \right\|_2^2 \tag{3.3}$$

En revanche, si le modèle est interpolant l'erreur commise sur les points du DOE est forcément nulle et la fonction objectif à optimiser a une forme différente. C'est par exemple le cas du krigeage pour lequel les paramètres sont choisis en calculant le maximum de vraisemblance, qui peut être écrit sous la forme de l'équation (3.2). Cette question est traitée plus en détail en section 3.2.3.

3.1.1 Compromis biais-variance pour la sélection de modèle

Afin de construire le modèle commettant l'erreur de prédiction la plus faible, il ne suffit pas de fixer la complexité et de résoudre l'équation (3.3). En effet, l'erreur de prédiction commise par le modèle dépend très fortement de sa complexité, ce qui rend le choix de cette complexité crucial. Afin d'illustrer cela, plaçons nous dans le cas d'un modèle à une seule sortie pour simplifier les écritures et détaillons l'erreur de prédiction commise en n'importe quel point du domaine $\boldsymbol{x} \in \mathcal{D}_x \subset \mathbb{R}^{N_x}$ par un modèle de substitution à complexité fixée C :

$$\mathcal{E}_P\left(\boldsymbol{x}\right) = \mathbb{E}\left[\left(\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) - f(\boldsymbol{x}) \right)^2 \,\middle|\, DOE \right]$$

Or $f(\boldsymbol{x})$ est la cible, considérée sans bruit. On a donc un problème du type, avec $a \in \mathbb{R}$:

$$\begin{aligned} \mathbb{E}\left[(Z-a)^2\right] &= \mathbb{E}\left[(Z - \mathbb{E}\left[Z\right] + \mathbb{E}\left[Z\right] - a)^2\right] \\ &= \mathbb{E}\left[(Z - \mathbb{E}\left[Z\right])^2\right] + (a - \mathbb{E}\left[Z\right])^2 \end{aligned}$$

Comme $\mathbb{E}[Z]$ et a sont des constantes, on a donc : $\mathbb{E}[(Z - \mathbb{E}[Z])(a - \mathbb{E}[Z])] = 0$. Ce qui donne l'équation (3.4), en omettant le *DOE*, sur lequel porte l'aléa, pour simplifier l'écriture.

$$\mathcal{E}_P(\boldsymbol{x}) = \underbrace{\mathbb{E}\left[\left(\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) - \mathbb{E}\left[\hat{f}(\boldsymbol{x};\ \boldsymbol{w})\right]\right)^2\right]}_{\text{Variance}} + \underbrace{\left(\mathbb{E}\left[\hat{f}(\boldsymbol{x};\ \boldsymbol{w})\right] - f(\boldsymbol{x})\right)^2}_{\text{Biais}^2} \tag{3.4}$$

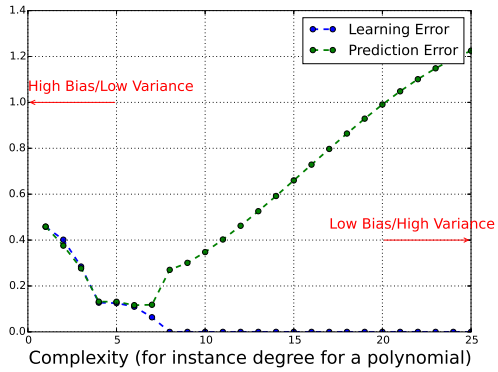


FIGURE 3.2 – Représentation de l'erreur de prédiction et de l'erreur d'apprentissage en fonction de la complexité

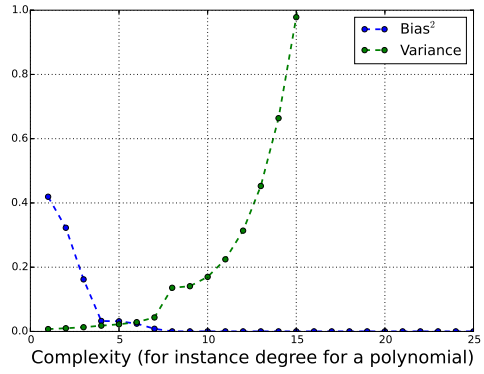


FIGURE 3.3 – Représentation du biais et de la variance en fonction de la complexité

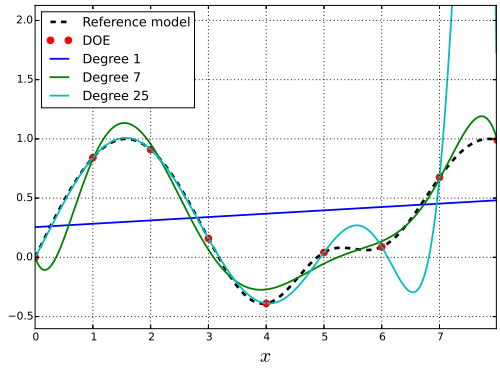


FIGURE 3.4 – Représentation de la fonction à approcher et de différentes régressions polynomiales avec le degré variant

Par exemple, le cas d'un polynôme en 1D est illustré en figure 3.4, donc avec $\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) = w_0 + \sum_{i=1}^{N_x} w_i x_i$ et les paramètres $\boldsymbol{w}$ choisis en résolvant l'équation (3.3). Sur la figure 3.3, on remarque que le biais est important lorsque le degré du polynôme est petit puisque l'erreur commise par le prédicteur est grande, même sur les points d'apprentissage. Par ailleurs, la variance est faible puisqu'il y a peu de degrés de liberté dans sa construction. En revanche, il n'est pas bon d'utiliser une complexité trop grande pour représenter une fonction puisque cela augmente la variance de la prédiction et l'on aboutit ainsi à un sur-apprentissage de la fonction, c'est-à-dire que le modèle de substitution va trop « coller » aux données du DOE. Ceci réduit sa capacité de généralisation. Les figures 3.2 et 3.4 illustrent ce phénomène. On voit que lorsque le polynôme utilisé a un degré trop élevé par rapport à la fonction à prédire, même si l'erreur d'apprentissage continue de décroître, l'erreur de prédiction grandit.

En fait, ceci traduit le fait que la complexité du modèle de substitution doit être adaptée à la complexité de la fonction à prédire et à la taille de l'échantillon d'apprentissage disponible. En effet, si la complexité du modèle réduit est trop faible, il y a un sous-

apprentissage, caractérisé par un biais élevé et une variance faible. En revanche, si elle est trop importante, il y a un sur-apprentissage des données du DOE avec un biais faible mais une variance importante.

Une problématique cruciale en réduction de modèle par apprentissage statistique apparaît ici : comment détecter la complexité optimale sachant que la seule erreur d'apprentissage ne mesure pas la capacité de généralisation du modèle ?

3.1.2 Présentation des méthodes de sélection de modèle

Afin de trouver le meilleur compromis entre le biais et la variance du modèle obtenu en faisant varier la complexité $C \in [\![C_{min}, C_{max}]\!]$, il est nécessaire d'avoir une mesure de l'erreur de généralisation assignée à chaque valeur de complexité. Par exemple dans la figure 3.3, cette erreur était estimée sur une base de test annexe.

Il existe notamment deux types de stratégies pour y parvenir :

1. Diviser le DOE en sous-ensembles disjoints $DOE = \mathcal{T} \oplus \mathcal{A}$ et $\mathcal{A} \cap \mathcal{T} = \emptyset$. L'un sert à tester (ensemble de test $\mathcal{T}$) les modèles construits à partir des échantillons de l'autre (ensemble d'apprentissage $\mathcal{A}$).
2. Pénaliser l'erreur d'apprentissage en faisant intervenir la complexité du modèle.

La première ne fait aucune hypothèse sur la complexité et ne regarde que l'effet de cette dernière sur la prédiction. En revanche, la division du DOE en sous-ensembles nécessite la présence de nombreux échantillons. En effet, si tel n'est pas le cas, alors le cardinal de $\mathcal{A}$ et de $\mathcal{T}$ est réduit. Or, un faible cardinal de $\mathcal{A}$ implique un biais élevé étant donné que peu d'échantillons d'apprentissage sont utilisés, et un faible cardinal de $\mathcal{T}$ implique une erreur de généralisation mal estimée ce qui implique que la complexité de la vraie fonction risque de ne pas être captée. À l'aide de simulations statistiques telles que la validation croisée [Efron, 1983] ou le *bootstrap* [Efron, 1979], il est possible d'estimer l'erreur de généralisation en gardant l'idée précédente mais avec un DOE de cardinal plus réduit.

Le second point consiste à rajouter un terme dans l'erreur de l'équation (3.3) de sorte à pénaliser la complexité du modèle, synonyme de variance élevée. Or ceci nécessite de pouvoir quantifier l'effet de la complexité sur la variance. Cette quantification repose sur l'introduction d'hypothèses, ce qui se traduit par l'existence de différents critères, chacun ayant un effet différent sur les paramètres $\boldsymbol{w}$.

3.1.2.1 Sélection par simulation de l'erreur de généralisation

La première manière de faire est de calculer, parallèlement à la minimisation de l'erreur d'apprentissage, une estimation de l'erreur de généralisation à l'aide d'échantillons écartés de l'étape de minimisation des paramètres $\boldsymbol{w}$.

Estimateur *hold-out* : Une première manière d'approcher cette erreur est de diviser le DOE en deux sous-ensembles $DOE = \mathcal{T} \oplus \mathcal{A}$ et $\mathcal{A} \cap \mathcal{T} = \emptyset$. L'ensemble $\mathcal{A}$ permet de minimiser l'erreur d'apprentissage RMSE 3.3 et l'ensemble $\mathcal{T}$ permet quant à lui d'estimer l'erreur de généralisation. L'estimateur induit, appelé estimateur *hold-out*, est calculé par la RMSE commise sur les points de $\mathcal{T}$:

$$\mathcal{E}_P(C) = \mathcal{E}(\hat{f}\,(\bullet;\ \boldsymbol{w})\,, \mathcal{T}) = \sum_{x_i \in \mathcal{T}} \left\| \hat{f}(\boldsymbol{x}^i;\ \boldsymbol{w}) - f(\boldsymbol{x}^i) \right\|^2 \text{ avec } N_{\boldsymbol{w}} = \mathcal{C}\,(C, N_x, N_y) \quad (3.5)$$

Algorithme 3.1 Sélection de modèle par estimateur *hold-out*

Input : $DOE = \left\{(\boldsymbol{x}^i, \boldsymbol{y}^i = f(\boldsymbol{x}^i))_{i\in[1,N_{DOE}]}\right\}$
La famille de modèles analytiques $\hat{f}(\bullet;\ \boldsymbol{w})$
Output : La complexité optimale C_{opt}
· Diviser $DOE = \mathcal{T} \oplus \mathcal{A}$;
for $C \in [\![C_{min}, C_{max}]\!]$ **do**
· Calculer $\boldsymbol{w}$ en minimisant $\mathcal{E}(\hat{f}\,(\bullet;\ \boldsymbol{w})\,, \mathcal{A})$ cf équation (3.3);
· Calculer l'estimation de l'erreur de généralisation sur l'ensemble de test $\mathcal{E}_P(C)$ cf équation (3.5);
end
· Choisir $C_{opt} = \text{argmin}_C\ \mathcal{E}_P(C)$;

Afin que cet estimateur soit fiable, il est nécessaire que le cardinal de DOE soit suffisamment grand. De plus, il est également nécessaire de garder le cardinal de $\mathcal{A}$ suffisamment élevé de sorte à minimiser le biais du modèle. Or, du fait du coût important d'exécution du vrai modèle f, il arrive très souvent que la taille du DOE soit réduite, ce qui rend impossible l'utilisation du *hold-out*.

Estimateur par validation croisée : Afin de rentabiliser tous les échantillons disponibles lorsqu'il y en a peu, c'est-à-dire lorsque le cardinal du DOE N_{DOE} est réduit, une manière de faire est d'utiliser la méthode de validation croisée. Son principe, très utilisé en statistique [Efron, 1983] [Lecué *et al.*, 2012], conserve l'idée du *hold-out* en partitionnant le DOE en V sous-ensembles $\{\mathcal{V}_i\}_{i\in[\![1,V]\!]}$ appelés *folds* donc $DOE = \bigoplus_{i=1}^{V} \mathcal{V}_i$. Ces sous-ensembles peuvent être disjoints, donc $\forall i, j \in [\![1, V]\!]$, $i \neq j \Rightarrow \mathcal{V}_i \cap \mathcal{V}_j = \emptyset$, ou non.

Algorithme 3.2 Sélection de modèle par validation croisée

Input : $DOE = \left\{(\boldsymbol{x}^i, \boldsymbol{y}^i = f(\boldsymbol{x}^i))_{i\in[1,N_{DOE}]}\right\}$
La famille de modèles analytiques $\hat{f}(\bullet;\ \boldsymbol{w}^i)$
Le nombre de *folds* V
Output : La complexité optimale C_{opt}
· Diviser $DOE = \bigoplus_{i=1}^{V} \mathcal{V}_i$;
for $C \in [\![C_{min}, C_{max}]\!]$ **do**
for $i \in [\![1, V]\!]$ **do**
· $\mathcal{A} = \bigoplus_{j\neq i} \mathcal{V}_j$ et $\mathcal{T} = \mathcal{V}_i$;
· Calculer $\boldsymbol{w}^i$ en minimisant $\mathcal{E}(\hat{f}\,(\bullet;\ \boldsymbol{w})\,, \mathcal{A})$ cf équation (3.3);
· Calculer l'estimation de l'erreur sur l'ensemble de test $\mathcal{E}_i = \mathcal{E}(\hat{f}\,(\bullet;\ \boldsymbol{w}^i)\,, \mathcal{T})$;
end
· Calculer l'estimation de l'erreur de généralisation $\hat{\mathcal{E}}_P(C) = \frac{1}{V}\sum_{i=1}^{V} \mathcal{E}_i$;
end
· Choisir $C_{opt} = \text{argmin}_C\ \hat{\mathcal{E}}_P(C)$;

Pour que tous les échantillons du DOE servent à minimiser l'erreur commise par les paramètres $\boldsymbol{w}$ en résolvant l'équation (3.3), l'idée est de construire un modèle avec $V - 1$

des sous-ensembles, puis de tester ce modèle sur le $V^{\text{ième}}$ sous-ensemble. En calculant l'erreur commise sur chacun des sous-ensembles de test, on obtient donc V erreurs de tests. L'estimation de l'erreur de généralisation est donc la moyenne de ces V erreurs de test. L'algorithme 3.2 détaille les étapes de cette estimation. En ce qui concerne le choix de V, un V trop petit met de côté beaucoup d'échantillons pour chaque apprentissage. À l'inverse, un V trop grand demande un temps de calcul très important dans le cas où la construction du modèle est plus coûteuse que son exécution. Si V est maximal, c'est-à-dire $V = \text{card}\,(DOE)$, on appelle cette technique le *leave-one-out*. Elle présente notamment l'avantage de ne pas avoir à choisir une répartition des échantillons dans les *folds*, sachant que la variabilité du résultat par rapport à cette répartition peut-être importante. Bien qu'il existe un V optimal, sa recherche est coûteuse et en pratique un choix très répandu se trouve entre 5 et 10 [Dreyfus *et al.*, 2002], selon le budget de calcul.

Estimateur par *bootstrap* : Le *bootstrap* [Efron, 1979] est une autre alternative au *hold-out* utilisant tous les échantillons du DOE dans le but d'estimer l'erreur de généralisation.

Le principe du *bootstrap* est le suivant : il consiste en la répétition de l'apprentissage $B \in \mathbb{N}$ fois sur les B ensembles d'apprentissage $(\mathcal{A}_b)_{b\in[\![1,B]\!]}$ construits à partir du DOE. Plus précisément, ces B sous-ensembles sont générés par N_B tirages avec remise dans l'ensemble DOE. Sur chacun des B ensembles d'apprentissage $\left(\mathcal{A}^b\right)_{b\in[\![1,B]\!]}$ de N_B échantillons, on minimise l'erreur d'apprentissage pour trouver les paramètres optimaux correspondants $\boldsymbol{w}^b = \text{argmin}_{v\in\mathbb{R}^{N_w}}\ \mathcal{E}\left(\hat{f}\left(\bullet;\ \boldsymbol{v}\right), \mathcal{A}^b\right)$.

L'estimation de l'erreur de généralisation par bootstrap est donc, à complexité fixée C :

$$\mathcal{E}_P(C) = \frac{1}{B}\sum_{b=1}^{B} \mathcal{E}\left(\hat{f}\left(\bullet;\ \boldsymbol{w}^b\right), DOE\right) \quad \text{avec} \quad \begin{cases} \boldsymbol{w}^b = \underset{v\in\mathbb{R}^{N_w}}{\text{argmin}} \mathcal{E}\left(\hat{f}\left(\bullet;\ \boldsymbol{v}\right), \mathcal{A}^b\right) \\ N_w = \mathcal{C}\left(C, N_x, N_y\right) \end{cases} \tag{3.6}$$

L'algorithme 3.3 résume cette technique.

Algorithme 3.3 Sélection de modèle par bootstrap

Input : $DOE = \left\{(\boldsymbol{x}^i, \boldsymbol{y}^i = f(\boldsymbol{x}^i))_{i\in[1,N_{DOE}]}\right\}$
La famille de modèles analytiques $\hat{f}\left(\bullet;\ \boldsymbol{w}\right)$
Le nombre de répétitions bootstrap B et de tirages avec remise N_b
Output : La complexité optimale C_{opt}

· Générer les B sous-ensembles de DOE par N_b tirages avec remise $\left(\mathcal{A}^b\right)_{b\in[\![1,B]\!]}$;
for $C \in [\![C_{min}, C_{max}]\!]$ **do**
  for $b \in [\![1, B]\!]$ **do**
    · Calculer $\boldsymbol{w}^b$ en minimisant $\mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), \mathcal{A}^b)$ cf équation (3.3);
    · Calculer l'estimation de l'erreur sur DOE $\mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}^b\right), DOE)$;
  end
  · Calculer l'estimation de l'erreur de généralisation $\hat{\mathcal{E}}_P(C)$ cf équation (3.6);
end
· Choisir $C_{opt} = \text{argmin}_C\ \hat{\mathcal{E}}_P(C)$;

Remarque sur l'utilisation du bootstrap pour estimer la variance de l'estimation du modèle en un point $\boldsymbol{x} \in \mathcal{D}_x$: L'un des avantages du *bootstrap* est qu'il permet également d'approcher la distribution de la sortie du modèle de substitution $\hat{f}(\boldsymbol{x};\ \boldsymbol{w})$ en chaque $\boldsymbol{x} \in \mathcal{D}_x$. Ceci se fait à l'aide de la distribution empirique calculée sur les B modèles construits, ici notée $\hat{F}^B(\boldsymbol{x})$. En calculant la variance de cette distribution empirique, on obtient un estimateur asymptotiquement convergent de la variance de la sortie du modèle de substitution $\hat{f}(\boldsymbol{x};\ \boldsymbol{w})$ en chaque $\boldsymbol{x} \in \mathcal{D}_x$, appelée $\hat{\sigma}(\boldsymbol{x})$:

$$\begin{aligned} \mathbb{E}\left[\hat{F}^B(\boldsymbol{x})\right] &= \frac{1}{B}\sum_{b=1}^{B} \hat{f}(\boldsymbol{x};\ \boldsymbol{w}^b) \\ \hat{\sigma}(\boldsymbol{x}) &= \mathrm{Var}\left[\hat{F}^B(\boldsymbol{x})\right] = \frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{f}(\boldsymbol{x};\ \boldsymbol{w}^b) - \mathbb{E}\left[\hat{F}^B(\boldsymbol{x})\right]\right)^2 \end{aligned} \tag{3.7}$$

3.1.2.2 Sélection par pénalisation

La figure 3.3 montre que l'erreur d'apprentissage calculée à partir de l'équation (3.3) ne mesure pas la capacité de généralisation du modèle construit. Or, l'objectif est bien d'obtenir le modèle dont la complexité lui permet d'avoir la meilleure capacité de généralisation possible. De plus, la figure 3.3 montre également que ce qui fait perdre la capacité de généralisation du modèle, c'est l'augmentation de variance due à la complexité trop importante du modèle.

Une manière de choisir le modèle le plus efficace en prédiction serait donc de prendre en compte sa complexité dans le calcul des paramètres $\boldsymbol{w}$. Pour cela, on peut en particulier modifier le problème d'optimisation à résoudre de l'équation (3.2) en pénalisant les modèles de complexité trop importante à l'aide d'une fonction de pénalisation $\mathrm{Pen}(\boldsymbol{w})$:

$$\begin{array}{rccc} \mathrm{Pen} & : & \mathbb{R}^{N_w} & \longrightarrow & \mathbb{R} \\ & & \boldsymbol{w} & \longmapsto & \mathrm{Pen}(\boldsymbol{w}) \end{array}$$

$$\mathcal{E}^{pen}(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE) = \mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE) + \mathrm{Pen}(\boldsymbol{w}) \tag{3.8}$$

Cette fonction de pénalisation doit être croissante avec la dimension N_w de ces paramètres. L'utilisation de cette fonction de pénalisation pour le choix de la complexité est donnée à l'algorithme 3.4. Il existe plusieurs manières de définir une telle fonction de pénalisation. Une manière simple, et très répandue par exemple dans le domaine des problèmes inverses car permettant d'améliorer le conditionnement d'un problème mal posé, est la régularisation appelée *ridge regression* ou régularisation de Tikhonov. Elle consiste à prendre un terme qui pénalise les paramètres ayant une norme 2 trop élevée, donc :

$$\mathrm{Pen}(\boldsymbol{w}) = \lambda \left\|\boldsymbol{w}\right\|_2^2 = \lambda \sum_{i=1}^{N_w} w_i^2 \tag{3.9}$$

Cette fonction de pénalisation a donc le double avantage de pénaliser les complexités trop élevées et de régulariser le problème d'optimisation à résoudre, le rendant plus stable et donc plus facile à résoudre [Tarantola, 2004].

En gardant le même principe et en l'élargissant à d'autres normes, on peut définir bien d'autres fonctions de pénalisation en se permettant de modifier la valeur de l'exposant que

l'on applique aux paramètres :

$$\text{Pen}(\boldsymbol{w}) = \lambda \sum_{i=1}^{N_w} |w_i|^q \tag{3.10}$$

L'une des fonctions également répandue en statistique est la régularisation utilisant $q = 1$. Dans la littérature, elle est le plus souvent identifiée sous le nom de Lasso (pour *Least Absolute Shrinkage and Selection Operator*) [Tibshirani, 1994]. L'avantage qu'a la régression Lasso par rapport à la *ridge regression* est qu'elle force certains des paramètres $\boldsymbol{w}$ à avoir une valeur nulle, ce que ne permettait pas le cas $q = 2$ où la valeur des w_i tend vers 0 plus continûment, sans toutefois l'atteindre. En revanche, la valeur absolue introduite par la norme 1 rend la fonction coût à minimiser non différentiable, ce qui peut complexifier le problème d'optimisation à résoudre, même si des algorithmes d'optimisation spécifiques existent tels que LARS [Efron *et al.*, 2004].

Choix du λ Une des difficultés de ces régularisations est de fixer la valeur du λ dans l'équation (3.9). En effet, une valeur trop petite de ce paramètre rend le paramètre de pénalisation négligeable et donc inutile. À l'inverse, une valeur trop importante de ce paramètre implique un trop grand éloignement au problème initial à résoudre, ce qui peut en particulier introduire un biais sur la prédiction. Afin de trouver le λ optimal, une manière très couramment utilisée en apprentissage statistique est d'utiliser la validation croisée introduite précédemment, en testant différentes valeurs pour ce paramètre.

Algorithme 3.4 Sélection de modèle par pénalisation

Input : $DOE = \left\{(\boldsymbol{x}^i, \boldsymbol{y}^i = f(\boldsymbol{x}^i))_{i\in[1,N_{DOE}]}\right\}$
La famille de modèles analytiques $\hat{f}(\bullet;\ \boldsymbol{w})$
La fonction de pénalisation Pen
Output : La complexité optimale C_{opt}
for $C \in [\![C_{min}, C_{max}]\!]$ **do**
· Calculer $\boldsymbol{w}^C$ les paramètres associés à la complexité C en minimisant $\mathcal{E}^{pen}(\hat{f}\left(\bullet;\ \boldsymbol{w}^C\right), DOE)$ cf équation (3.8);
end
· Choisir $C_{opt} = \text{argmin}_C\ \mathcal{E}^{pen}(\hat{f}\left(\bullet;\ \boldsymbol{w}^C\right), DOE)$;

Autres critères de pénalisation : D'autres critères, basés sur le maximum de vraisemblance, permettent également de pénaliser la complexité (AIC, BIC ...). Le lecteur intéressé par ces critères peut se référer à [Hastie *et al.*, 2009].

3.1.3 Génération du plan d'expériences

Jusqu'à présent, rien n'a été dit sur la manière de choisir les entrées $\boldsymbol{X} = (\boldsymbol{x}^i)_{i\in[1,N_{DOE}]}$ sur lesquelles la fonction à substituer f doit être estimée. Or, il est évident que ce choix peut avoir une influence importante sur la prédiction. En effet, dans le cas extrême où tous les points sont tirés dans un même sous-espace de $\mathcal{D}_x$, le modèle de substitution $\hat{f}$ aura une erreur très grande loin de cette zone. La distance entre les points du DOE est donc

un paramètre très important à prendre en compte lors de sa génération. Il correspond en fait à une mesure de discrétisation de l'espace $\mathcal{D}_x$.

Un autre paramètre important dans la construction d'un DOE est le budget de calcul N_{DOE} qui peut être réalisé sur la vraie fonction f. Par exemple, dans le cas d'un budget très restreint et si la dimension de $\boldsymbol{x}$ est grande, les méthodes déterministes tels que les plans factoriels deviennent rapidement inutilisables.

Enfin, dernier paramètre pouvant orienter la construction du plan : le modèle de substitution à construire fournit-il une erreur de prédiction en plus de la prédiction elle-même ? Si tel est le cas, un premier modèle peut être construit à partir d'une méthode d'échantillonnage *a priori*. Puis, à l'aide de l'erreur de ce modèle, des méthodes existent afin d'enrichir séquentiellement le DOE.

Dans un premier temps, sachant que les méthodes d'échantillonnage *a priori* sont indispensables, ces méthodes sont présentées en détaillant dans quel cas chacune d'entre elles peut être intéressante. Dans un second temps, les méthodes d'enrichissement de plan d'expériences basées sur les erreurs de prédiction du modèle sont expliquées : ceci est détaillé en section 3.2.4.

Échantillonnage *a priori*

Afin de pouvoir construire un premier modèle, il faut d'abord avoir accès à un DOE initial que l'on doit générer *a priori*, c'est-à-dire sans informations sur la fonction à prédire. Une première méthode intuitive serait de tirer les points aléatoirement selon une loi uniforme dans l'espace des entrées $\mathcal{D}_x$. Le problème de cette technique est qu'elle ne garantit aucunement que ces points soient répartis de manière optimale, c'est-à-dire qu'ils remplissent au mieux l'espace des entrées. Ceci est illustré en figure 3.5.

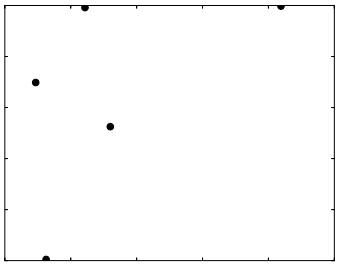


FIGURE 3.5 – Exemple d'un plan d'expériences aléatoire en dimension 2

Afin de s'assurer de cela, une première manière, déterministe et simple à mettre en œuvre, est de placer les points sur une grille en choisissant un nombre de niveau n_{niv} par dimension. Or, le nombre de points à placer augmente rapidement avec le nombre de dimensions $N_{DOE} = n_{niv}^{N_x}$. Par exemple, avec seulement 4 points par dimension, donc $n_{niv} = 4$, la taille du plan d'expériences excède les 100 points dès la dimension $N_x = 4$.

Autre méthode, permettant de générer un DOE à budget fixé et plus répandue dans le domaine de l'apprentissage statistique est la génération de plans d'expériences par *Latin Hypercube Sampling* (LHS) [Viana *et al.*, 2009]. Afin de garantir la non-proximité de deux points du DOE, il divise l'espace $\mathcal{D}_x$ en hypercubes disjoints avec une grille (cf figure 3.7) de N_{DOE} sous-intervalles par dimension. Les points sont ensuite tirés de sorte à avoir un unique point par colonne et par ligne. De cette manière, le nombre de points N_{DOE} à générer définit en quelque sorte une taille de maille, à partir de laquelle les points sont

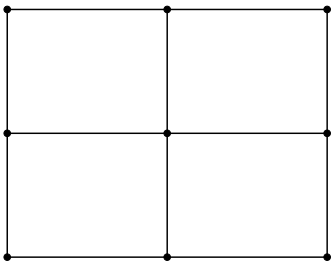


FIGURE 3.6 – Exemple d'un plan d'expériences factoriel à trois niveaux en dimension 2, donc $3^2 = 9$ points

générés. Par conséquent, quel que soit le budget alloué, tout l'espace est rempli de manière plus ordonnée que dans le cas totalement aléatoire et ce sans faire exploser le nombre de points nécessaires comme dans les plans factoriels. Par ailleurs, la figure 3.7 illustre le cas équiprobable où aucune zone n'est privilégiée par rapport à une autre, donc à taille de maille constante. Mais il est tout à fait envisageable de définir une largeur de maille variable, permettant de discrétiser plus finement certaines zones par rapport à d'autres.

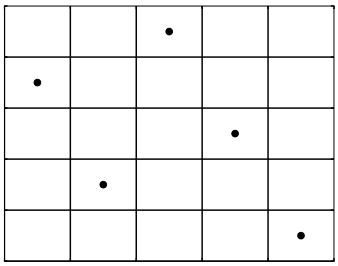


FIGURE 3.7 – Exemple d'un plan d'expériences LHS en dimension 2

Enfin, il existe des plans construits séquentiellement afin de garantir une discrépance faible. La discrépance est une mesure de l'écart entre une distribution de points uniforme avec la distribution de points proposée. Elle mesure donc l'irrégularité de la distribution. La construction de plans à discrépance faible est basée sur la construction séquentielle du DOE de sorte à couvrir l'espace de manière la plus efficiente possible en utilisant un critère d'optimalité donné [Kocis et Whiten, 1997] [Franco, 2008]. L'article de Pronzato et Müller [Pronzato et Müller, 2012] donne les détails de ces critères d'optimalité. En d'autres termes, elles permettent le remplissage séquentiel de l'espace sans connaissance de la fonction à prédire. On leur préfère donc l'utilisation de critères basés sur la variance du modèle de substitution construit à partir d'un plan LHS. Ces techniques sont présentées en section 3.2.4.

3.2 Construction de modèles de substitution de codes stationnaires

3.2.1 Présentation générale des différents types de modèle de substitution existants

Il existe dans la littérature une grande diversité de modèles de substitution, en d'autres termes de formes analytiques différentes pour $\hat{f}$ à laquelle on peut associer une erreur à minimiser $\mathcal{E}$. L'étude de Wang [Wang et Shan, 2007] propose une revue des modèles de substitution existants, parmi lesquels on peut citer les RBF (pour *Radial Basis Function*) [Park et Sandberg, 1991] [Buhmann, 2003] qui sont des réseaux de neurones utilisant une fonction d'activation particulière, les SVR pour *Support Vector Regression* [Basak *et al.*, 2007] [Hastie *et al.*, 2009, pp.434-438] qui sont une généralisation à la régression d'une technique de classification ou encore les splines [Hastie *et al.*, 2009, pp.139-190] qui sont des régressions polynomiales par morceaux. Nous faisons ici le choix de nous limiter à deux techniques représentatives des méthodes les plus répandues : les réseaux de neurones artificiels en section 3.2.2 comme exemple de modèle régressant et les processus gaussiens ou krigeage en section 3.2.3 comme exemple de modèle interpolant.

3.2.2 Un exemple de modèle régressant : Les réseaux de neurones artificiels

Les réseaux de neurones proposent une formulation analytique non-linéaire afin d'exprimer une fonction à $N_y \geqslant 1$ sorties $\hat{f} = \left(\hat{f}_1, \cdots, \hat{f}_{N_y}\right)$. Leur conception est fortement inspirée du fonctionnement des neurones biologiques. En effet, un neurone artificiel peut être vu comme une fonction de transfert permettant de transformer ses entrées $\boldsymbol{e} = (e_1, \cdots, e_{N_e}) \in \mathbb{R}^{N_e}$ en une sortie $z \in \mathbb{R}$ selon des règles précises, dépendant de l'utilisation du neurone. Leur domaine d'application étant très vaste, cette partie ne se focalise que sur l'utilisation des réseaux de neurones à des fins d'approximation de fonctions. Dans ce cadre, un neurone s'écrit :

$$z(\boldsymbol{e}) = \phi\left(w_0 + \sum_{i=1}^{N_e} w_i e_i\right) \tag{3.11}$$

$\boldsymbol{w} = (w_0, \cdots, w_{N_e})$ sont les poids permettant de faire une combinaison linéaire des entrées.

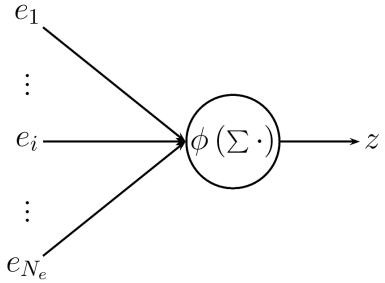


FIGURE 3.8 – Schéma d'un neurone artificiel

ϕ peut prendre les formes suivantes (illustrées en figure 3.9) :

— Fonction de Heaviside :

$$\phi(x) = \begin{cases} 0 \text{ si } x < 0 \\ 1 \text{ si } x \geqslant 0 \end{cases}$$

Étant donné sa forme non continue, elle n'est pas particulièrement adaptée à la régression et est plus classiquement utilisée en classification.

— Sigmoïde :

$$\phi(x) = \frac{1}{1 + \exp(-kx)}$$

avec $k > 0$ un paramètre de forme permettant de modifier la pente en $x = 0$ de cette fonction, et donc de s'éloigner ou de se rapprocher de la fonction de Heaviside précédente. C'est cette fonction qui est la plus utilisée en régression.

— Linéaire, c'est-à-dire $\phi(x) = x$. Cette fonction d'activation permet de se ramener à un modèle linéaire, les réseaux de neurones étant une généralisation de la régression linéaire.

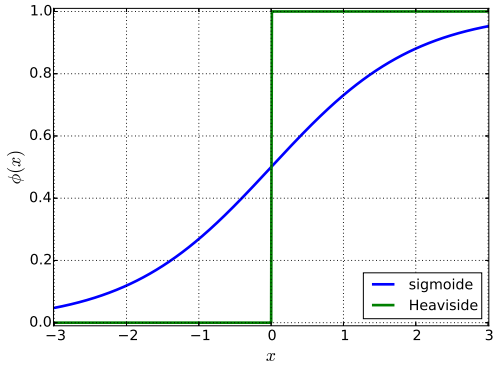


FIGURE 3.9 – Illustration des formes courantes pour la fonction d'activation ϕ. La sigmoïde est illustrée dans le cas où $k = 1$.

L'organisation du réseau en couches (cf figure 3.10) assure que toute fonction peut être prédite avec la précision souhaitée [Cybenko, 1989] [Barron, 1993]. C'est ce que l'on appelle un perceptron multicouches [Dreyfus *et al.*, 2002]. La forme à une couche d'entrée, une couche cachée et une couche de sortie, la plus classiquement utilisée pour le moindre nombre de paramètres à gérer, suffit d'ailleurs pour garder la propriété d'approximateur universel [Hornik, 1991]. Sa formule analytique est la suivante :

$$\begin{cases} \hat{f}_j(\boldsymbol{x};\ \boldsymbol{w}) &= \displaystyle\sum_{n=1}^{C} w''_{n,j} z_n(\boldsymbol{x}) + w''_{0,j} \\ z_n(\boldsymbol{x}) &= \phi\left(\displaystyle\sum_{i=1}^{N_x} w'_{i,n} x_i + w'_{0,n}\right) \end{cases} \tag{3.12}$$

Dans le cas d'un réseau de neurones, $\boldsymbol{w} = \{\boldsymbol{w}', \boldsymbol{w}''\} \in \mathbb{R}^{N_{w'}} \times \mathbb{R}^{N_{w''}}$ et la complexité C à estimer lors de l'apprentissage correspond au nombre de neurones. La formule $\mathcal{C}$ (cf équation (3.1)) permettant de calculer le nombre de poids à partir de cette complexité est la suivante :

$$N_w = \mathcal{C}(C, N_x, N_y) = \underbrace{C(N_x + 1)}_{N_{w'}} + \underbrace{(C + 1) N_y}_{N_{w''}} \tag{3.13}$$

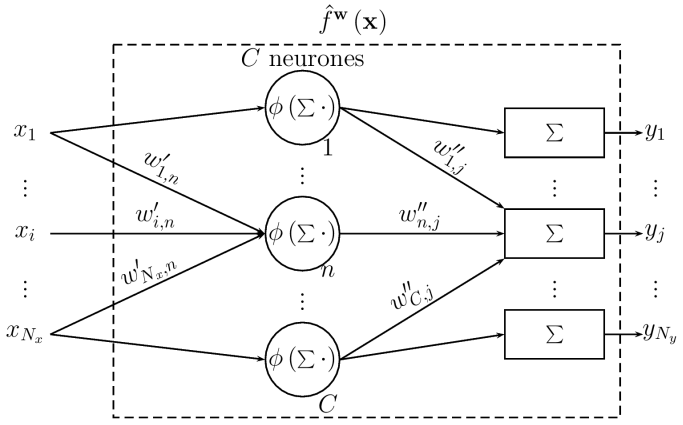


FIGURE 3.10 – Schéma d'un perceptron multicouches avec une couche d'entrée, une couche cachée et une couche de sortie - les biais $w_{0,\bullet}$ ne sont pas représentés

En revanche, les perceptrons multicouches rendent difficile toute signification physique. En d'autres termes, ce n'est qu'une base qui permet d'approcher n'importe quelle fonction mais en perdant toute information sur le lien entre les entrées et les sorties.

3.2.2.1 Apprentissage d'un perceptron multicouches

Comme détaillé précédemment, l'apprentissage d'un perceptron multicouches se fait en minimisant l'erreur quadratique moyenne commise par le modèle $\hat{f}$ sur les points du DOE (cf équation (3.3)), avec éventuellement un terme de régularisation (cf section 3.1.2.2). En revanche, la résolution de ce problème d'optimisation n'est pas facile : la fonction coût $\mathcal{E}\left(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE\right)$ est caractérisée par la présence de multiples minima locaux [Bishop, 2006] et plus la dimension N_w de ce problème d'optimisation est grande et plus le nombre de minima locaux peut être important. Le calcul des poids d'un réseau de neurones peut donc être coûteux en temps de calcul.

Les algorithmes d'optimisation les plus utilisés afin de trouver les poids optimaux sont basés sur des algorithmes par descente de gradient du premier voire du second ordre utilisant une approximation de la Hessienne tels que les algorithmes BFGS [Battiti et Masulli, 1990] ou Levenberg-Marquardt [Moré, 1978] [Hagan et Menhaj, 1994]. Levenberg-Marquardt est un algorithme particulièrement intéressant puisqu'il inclut dans sa résolution un terme de régularisation μ^{it} tendant vers 0 avec les itérations, le rendant particulièrement stable. Pour illustrer cela, voici le terme de descente à chaque itération de l'algorithme de Levenberg-Marquardt :

$$\left\{ \begin{array}{l} \boldsymbol{w}^{it+1} = \boldsymbol{w}^{it} + \alpha \boldsymbol{d}^{it} \\ \boldsymbol{d}^{it} = \left[\left(\nabla_{\boldsymbol{w}} \mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}^{it}\right), DOE) \right) \left(\nabla_{\boldsymbol{w}} \mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}^{it}\right), DOE) \right)^{\mathrm{T}} + \mu^{it} \underline{\underline{I_{N_w}}} \right]^{-1} \nabla_{\boldsymbol{w}} \mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}^{it}\right), DOE) \end{array} \right. \tag{3.14}$$

En revanche, l'un des inconvénients de l'utilisation des réseaux de neurones est la présence de multiples minima locaux à la fonction $\mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE)$ et sa forte non-linéarité, rendant la recherche du minimum global difficile. Sachant que les algorithmes basés sur le gradient sont des algorithmes locaux, c'est-à-dire trouvant le minimum local le plus proche du point de départ de l'optimisation, il est indispensable de réaliser cette

optimisation de nombreuses fois en partant de points d'initialisation différents. C'est ce que l'on appelle le *multistart*. Une manière très répandue d'initialiser les poids est proposée par l'étude de Nguyen et Widrow [Nguyen et Widrow, 1990].

Par ailleurs, du fait de la présence de nombreux minima locaux, d'autres études utilisent des algorithmes globaux stochastiques du type recuit simulé [Da et Xiurun, 2005], colonies de fourmis [Salama et Abdelbar, 2015], essaims particulaires [Wu et Chen, 2009], plus généralement évolutionnaires [Igel *et al.*, 2005] ou encore d'autres techniques sans gradient tels que Nelder-Mead [Liao *et al.*, 2015]. Toutefois, ces méthodes présentent quelques inconvénients : d'abord le nombre d'appels et de paramètres importants à gérer de ces méthodes rendent leur mise en place compliquée. De plus, il n'est pas important en apprentissage statistique de trouver le minimum global de l'erreur d'apprentissage : ceci peut mener au sur-apprentissage et donc à une erreur de prédiction importante.

3.2.2.2 Estimation du gradient par rétropropagation

La plupart des algorithmes d'apprentissage présentés précédemment nécessitent la connaissance du gradient de la fonction coût par rapport aux poids $\nabla_{\boldsymbol{w}}\mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}^{it}\right), DOE)$. Sachant que les réseaux de neurones fournissent une formulation analytique basée sur des sigmoïdes, qui sont infiniment différentiables, le gradient de la fonction coût par rapport aux poids est calculable analytiquement et de manière itérative par l'algorithme de rétropropagation du gradient [Rumelhart *et al.*, 1986]. Son principe repose essentiellement sur l'application répétée de la règle des dérivées composées. Il permet de calculer le gradient de la fonction coût suivante :

$$\mathcal{E}(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE) = \sum_{k=1}^{N_{DOE}} \left\| \hat{f}(\boldsymbol{x}^k;\ \boldsymbol{w}) - \boldsymbol{y}^k \right\|_2^2 = \sum_{k=1}^{N_{DOE}} \mathcal{E}^k$$

Cette erreur est décomposée en somme d'erreurs commises en chacun des échantillons $(\boldsymbol{x}^k, \boldsymbol{y}^k)$ du DOE. Pour chacune de ces composantes, le gradient est calculé de la manière suivante :

$$\frac{\partial \mathcal{E}^k}{\partial w_{ij}} = \left(\frac{\partial \mathcal{E}^k}{\partial v_i}\right)_k \left(\frac{\partial v_i}{\partial w_{ij}}\right)_k = \delta_i^k e_j^k \tag{3.15}$$

en omettant les $'$ et $''$ sur les poids w_{ij} et en considérant un neurone dont les entrées sont nommées e_i et avec $v_i = \sum_{j=1}^{N_e} w_{ij} e_j$ la somme des entrées pondérées du neurone i. Cette formule est composée de :

- $\left(\frac{\partial \mathcal{E}^k}{\partial z_i}\right)_k$ la valeur du gradient du coût partiel par rapport aux entrées pondérées lorsque les entrées du réseau sont celles qui correspondent à l'échantillon k.
- $\left(\frac{\partial v_i}{\partial w_{ij}}\right)_k$ la valeur de la dérivée partielle de cette combinaison des entrées par rapport au paramètre w_{ij} lorsque les entrées du réseau sont celles qui correspondent à l'échantillon k, ce qui explique que cela est égal à la valeur de l'entrée j du neurone i, e_j^k, lorsque les entrées du réseau sont celles qui correspondent à l'échantillon k.

Il ne reste donc à évaluer que les quantités δ_i^k traduisant le gradient du coût partiel par rapport aux entrées pondérées. Le nom de rétropropagation vient du calcul de ces composants. En effet, ils peuvent être avantageusement calculés de manière récursive en partant des sorties du réseau vers ses entrées :

— Pour un neurone de sortie i :

$$\delta_i^k = \left(\frac{\partial \mathcal{E}^k}{\partial z_i}\right)_k = -2\hat{f}(\boldsymbol{e}^k;\ \boldsymbol{w})$$

car la fonction d'activation des neurones de sortie est la fonction linéaire donc $\phi_s'(e) = 1$.

— Pour un neurone de la couche cachée i : la fonction de coût ne dépend des poids d'entrée de ce neurone caché que par l'intermédiaire des entrées des neurones de sortie m qui reçoivent la valeur de la sortie du neurone caché i. On a donc :

$$\delta_i^k = \sum_m \left(\frac{\partial \mathcal{E}^k}{\partial v_m}\right)_k \left(\frac{\partial v_m}{\partial v_i}\right)_k = \sum_m \delta_m^k \left(\frac{\partial v_m}{\partial v_i}\right)_k$$

Or $v_m = \sum_i w''_{mi} e_i^k = \sum_i w''_{mi}\phi(v_i^k)$ ce qui nous donne pour $\left(\frac{\partial v_m}{\partial v_i}\right)_k = w''_{mi}\phi'(v_i^k)$
On obtient donc finalement pour δ_i^k :

$$\delta_i^k = \sum_m \delta_m^k w''_{mi}\phi'(v_i^k) = \phi'(v_i^k)\sum_m \delta_m^k w''_{mi}$$

Il apparaît donc ici la récursivité du calcul de ces $\boldsymbol{\delta}$ puisqu'en parcourant le graphe des connexions depuis les sorties vers les entrées, le gradient des coûts partiels se développe efficacement. Ceci permet, en les sommant, d'obtenir le gradient de la fonction coût total.

Résumé de l'algorithme de rétropropagation : cet algorithme se décompose en deux phases pour chaque échantillon $k \in [\![1, N_{DOE}]\!]$:

1. La première phase de « propagation » où les entrées correspondant à l'échantillon k, $\boldsymbol{x}^k$, sont utilisées afin de calculer les sorties $\hat{\boldsymbol{y}}^k$ prédites par le réseau de neurones et donc les erreurs commises.
2. La deuxième phase de « rétropropagation » au cours de laquelle sont calculées les δ_i^k.

Une fois ces quantités évaluées, le gradient des fonctions de coût partiel peut être estimé par l'équation (3.15), puis le gradient du coût total $\frac{\partial \mathcal{E}(\hat{f}(\bullet;\ \boldsymbol{w}),DOE)}{\partial w_{ij}} = \sum_{k=1}^{N_{DOE}} \frac{\partial \mathcal{E}^k}{\partial w_{ij}}$

3.2.2.3 Sélection de modèle appliquée aux perceptron multicouches

Les réseaux de neurones étant des modèles régressants dont la construction se fait par minimisation de l'erreur moindres carrées de l'équation (3.3), les techniques de sélection de modèle détaillées en section 3.1.2 sont tout à fait applicables aux réseaux de neurones. Il existe cependant quelques modifications ou adaptations qui peuvent être faites. Notamment sur la *ridge regression*, une technique très répandue est de pénaliser différemment les poids de la couche d'entrée $\boldsymbol{w}'$ et ceux de la couche de sortie $\boldsymbol{w}''$ puisque leur rôle n'est pas le même. Comme il a été dit en section 3.1.2.2, ce genre de pénalisation favorise les solutions où la valeur des poids $\boldsymbol{w}$ n'est pas trop grande. Dans le cas des poids de la couche d'entrée des réseaux de neurones, ceci a pour effet de garder les combinaisons linéaires en entrée des fonctions d'activation sigmoïde au voisinage de leurs zones linéaires. Ce genre de régularisation permet donc d'obtenir des courbes prédites qui gagnent en régularité.

Par ailleurs, il existe une méthode permettant d'éviter le sur-apprentissage dans le cas des réseaux de neurones qui n'a pas été introduite précédemment car particulière à cette famille analytique. Il s'agit de l'*early stopping* ou arrêt prématuré. L'idée est d'arrêter l'algorithme d'optimisation des poids prématurément, c'est-à-dire avant convergence, afin que le modèle ne s'ajuste pas trop finement aux données d'apprentissage et ainsi éviter le sur-apprentissage. Cependant, il est important de ne pas arrêter trop précocement la minimisation de l'erreur sur le DOE puisque ceci peut introduire un biais d'estimation important. Une manière d'obtenir un critère d'arrêt satisfaisant est d'utiliser l'erreur sur une base de validation et d'arrêter les itérations lorsque cette dernière commence à croître. Ceci peut par ailleurs être aisément mis en place dans le cas de l'utilisation de méthodes de type *hold-out* ou validation croisée (cf section 3.1.2).

3.2.3 Un exemple de modèle interpolant : le krigeage

Le krigeage est un exemple de modèle interpolant très utilisé en optimisation [Jones *et al.*, 1998] [Forrester et Keane, 2009] ou encore en propagation d'incertitudes [Bect *et al.*, 2012]. Initialement développé pour la géostatistique [Krige, 1951] [Matheron, 1963], ce type de modèle a depuis été largement étendu à l'apprentissage statistique [Rasmussen et Williams, 2005]. Son principe est le suivant : il suppose que la fonction à approcher f est la réalisation d'un processus stochastique $Y(x)$, conditionné par les points du DOE. Cette hypothèse a notamment pour intérêt de fournir, en plus d'une prédiction de la fonction à approcher $\hat{f}$, un estimateur analytique de l'erreur commise par ce modèle de substitution $\hat{f}$ en tout point de l'espace $x \in \mathcal{D}_x$.

Dans cette partie, il est uniquement question du cas où f n'a qu'une seule sortie définie dans $\mathbb{R}$, donc $f \ : \ \mathcal{D}_x \longrightarrow \mathbb{R}$. Le cas où la fonction à prédire est à plusieurs sorties peut être traité en construisant un processus gaussien indépendant par sortie.

3.2.3.1 Approximation de fonction par krigeage

Le krigeage est la prédiction statistique d'une fonction en tout point de l'espace $x \in \mathcal{D}_x$ à partir d'une connaissance ponctuelle de la fonction fournie par le $DOE = \{(\boldsymbol{X}, \boldsymbol{Y})\}$ avec $\boldsymbol{X} = (\boldsymbol{x}^i)_{i\in[1,N_{DOE}]}$ et $\boldsymbol{Y} = (\boldsymbol{y}^i = f(\boldsymbol{x}^i))_{i\in[1,N_{DOE}]}$. Elle suppose pour cela que la fonction à prédire est la réalisation d'un processus gaussien Y composé d'une partie déterministe μ appelée aussi tendance et d'une partie aléatoire Z.

$$Y(\boldsymbol{x}) = \mu(\boldsymbol{x}) + Z(\boldsymbol{x}) \tag{3.16}$$

Ces deux fonctions μ et Z sont définies de la manière suivante :

— Le choix de la fonction μ détermine le type de krigeage utilisé :

Krigeage simple $\mu(x)$ est une constante connue, pouvant par exemple être prise nulle.

Krigeage ordinaire $\mu(x) = \mu \in \mathbb{R}$ une constante inconnue, à déterminer à partir des points du DOE.

Krigeage universel $\mu(x) = \sum_{k=0}^{N-1} w_k \phi_k(\boldsymbol{x})$ où $\phi = \{\phi_0, \cdots, \phi_{N-1}\}$ est une collection de fonctions de régression. Cette collection est le plus souvent une base de fonctions polynomiales.

Dans cette partie, il est surtout question du krigeage ordinaire. En effet, la partie déterministe n'est pas la partie la plus informative d'un modèle de krigeage, il n'est donc pas nécessaire d'y accorder trop d'efforts. Cependant, dans le cas où le nombre d'échantillons est très réduit ou dans le cadre de certaines applications, il peut être intéressant d'utiliser certaines formes particulières. Il existe d'ailleurs des méthodes combinant les polynômes de chaos, très répandus en propagation d'incertitudes, et le krigeage en utilisant le chaos polynomial comme tendance, dans le cadre d'un krigeage universel [Kersaudy *et al.*, 2015].

— Z est un processus gaussien stationnaire d'espérance nulle $\forall x \in \mathcal{D}_x$, $\mathbb{E}\left(Z(x)\right) = 0$, et de covariance :

$$\mathrm{Cov}(\boldsymbol{x}, \boldsymbol{x}') = \sigma_K^2 R(\boldsymbol{x}, \boldsymbol{x}';\ \boldsymbol{w}) \quad \forall (\boldsymbol{x}, \boldsymbol{x}') \in \mathcal{D}_x^2 \tag{3.17}$$

La fonction $R(\bullet, \bullet;\ \boldsymbol{w})\ :\ \mathcal{D}_x \times \mathcal{D}_x \longrightarrow \mathbb{R}$ est le noyau d'autocorrélation. Il permet de définir la corrélation spatiale entre deux points de l'espace des entrées. En effet, plus deux points sont proches et plus la fonction R doit être proche de 1. À l'inverse, plus deux points sont éloignés, plus la fonction R est proche de 0. Ce dernier est à définir par l'utilisateur et permet de prendre en compte les *a priori* disponibles sur la fonction à prédire. Par exemple si la fonction à prédire est très régulière, $f \in \mathcal{C}^\infty(\mathcal{D}_x, \mathbb{R})$, alors le noyau de corrélation gaussien est très approprié :

$$R(\boldsymbol{x}, \boldsymbol{x}';\ \boldsymbol{w}) = \exp\left(-\sum_{i=1}^{N_x}\left(\frac{x_i - x_i'}{w_i}\right)^2\right) \tag{3.18}$$

avec $\boldsymbol{w} = (w_1, \cdots, w_{N_x})$ les longueurs de corrélation qui représentent ici les paramètres à déterminer. Un autre exemple de l'*a priori* qui peut être défini par ce noyau est l'isotropie de la fonction f par rapport à ses entrées : si la fonction f est connue pour avoir la même dépendance envers chacune de ses entrées, on peut prendre $w_1 = \cdots = w_{N_x} = w$. Par ailleurs, l'hypothèse de stationnarité qui suppose qu'il existe une fonction r telle que $\mathrm{Cov}(Z(\boldsymbol{x}), Z(\boldsymbol{x}+\boldsymbol{h})) = r(\boldsymbol{h})$, permet de ne pas se soucier de la zone où l'on se trouve dans la formule de ce noyau R.

Grâce à l'hypothèse d'une sortie modélisée par processus gaussien, il est ensuite facile de déterminer la prédiction par krigeage de la fonction f en prenant l'espérance de ce processus gaussien, conditionné par le DOE :

$$\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) = \mathbb{E}\left[Y(\boldsymbol{x})\ \middle|\ \boldsymbol{Y} = f(\boldsymbol{X})\right] = \mu + \gamma(\boldsymbol{x})^T \boldsymbol{\Gamma}^{-1}\left(\boldsymbol{Y} - \mu\boldsymbol{1}\right) \tag{3.19}$$

avec :

— $\gamma\ :\ \mathcal{D}_x \longrightarrow \mathbb{R}^{N_{DOE}}$ représentant le vecteur d'autocorrélation entre le point $\boldsymbol{x}$ à prédire et les points du DOE.

$$\gamma(\boldsymbol{x}) = \Big(R(\boldsymbol{x}, \boldsymbol{x}^1;\ \boldsymbol{w}), \cdots, R(\boldsymbol{x}, \boldsymbol{x}^{N_{DOE}};\ \boldsymbol{w})\Big) \tag{3.20}$$

— $\boldsymbol{\Gamma} \in \mathcal{M}_{N_{DOE}}(\mathbb{R})$ la matrice des corrélations associée au DOE :

$$\boldsymbol{\Gamma} = \Big(R(\boldsymbol{x}^i, \boldsymbol{x}^j;\ \boldsymbol{w})\Big)_{1 \leqslant i,j \leqslant N_{DOE}} \tag{3.21}$$

La prédiction par krigeage est donnée par l'espérance, mais d'autres mesures peuvent être estimées analytiquement sur un processus gaussien comme sa variance en tout point, $\hat{\sigma}(\boldsymbol{x})$.

Cette variance peut notamment être utilisée comme une erreur de prédiction $\forall \boldsymbol{x} \in \mathcal{D}_x$, et elle vaut :

$$\hat{\sigma}(\boldsymbol{x})^2 = \mathrm{Var}\left[Y(\boldsymbol{x}) \mid \boldsymbol{Y} = f(\boldsymbol{X})\right] = \sigma_K^2 \left(1 - \gamma(\boldsymbol{x})^T \boldsymbol{\Gamma}^{-1} \gamma(\boldsymbol{x}) + \frac{1 - \mathbf{1}^T \boldsymbol{\Gamma}^{-1} \gamma(\boldsymbol{x})}{\mathbf{1}^T \boldsymbol{\Gamma}^{-1} \mathbf{1}}\right) \quad (3.22)$$

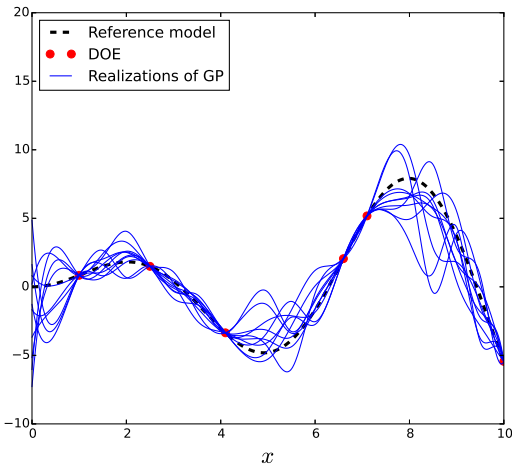


FIGURE 3.11 – Représentation de différentes réalisations d'un processus gaussien

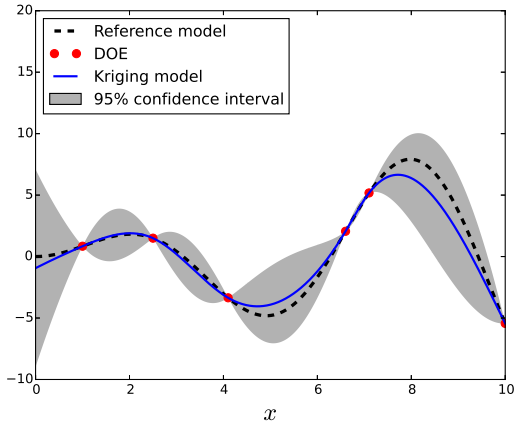


FIGURE 3.12 – Représentation de la prédiction par krigeage avec l'intervalle de confiance à 95%

Grâce à l'hypothèse gaussienne sur la loi de la sortie en chaque point $\boldsymbol{x} \in \mathcal{D}_x$, l'intervalle de confiance à 95% peut être déduit analytiquement des équations (3.19) et (3.22) :

$$IC^{0.95}(\boldsymbol{x}) \simeq \left[\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) - 1.96\hat{\sigma}(\boldsymbol{x}), \hat{f}(\boldsymbol{x};\ \boldsymbol{w}) + 1.96\hat{\sigma}(\boldsymbol{x})\right]$$

3.2.3.2 Calcul des longueurs d'autocorrélation

Les paramètres μ, $\boldsymbol{w}$ et la variance σ_K^2 sont le plus couramment estimés par maximum de vraisemblance sur les points du DOE. Les estimateurs de μ et σ_K^2 s'écrivent analytiquement. En ce qui concerne μ, sa valeur optimale correspond à la solution d'une approximation linéaire déterminée par moindres carrés :

$$\mu = \frac{\mathbf{1}^T \boldsymbol{\Gamma}(\boldsymbol{w})^{-1} \boldsymbol{Y}}{\mathbf{1}^T \boldsymbol{\Gamma}(\boldsymbol{w})^{-1} \mathbf{1}}$$

En ce qui concerne $\sigma_K^2(\boldsymbol{w})$, la solution est une formule analytique dépendant de $\boldsymbol{w}$ (en faisant apparaître les dépendances en $\boldsymbol{w}$ détaillées aux équations (3.17) et (3.21)) :

$$\sigma_K^2(\boldsymbol{w}) = \frac{1}{N_{DOE}} \left[\boldsymbol{Y} - \mu \mathbf{1}\right]^T \boldsymbol{\Gamma}(\boldsymbol{w})^{-1} \left[\boldsymbol{Y} - \mu \mathbf{1}\right]$$

Enfin, sachant que Y est un processus gaussien et en travaillant avec la log-vraisemblance [Marrel *et al.*, 2008], on arrive à la minimisation de la fonction suivante afin de trouver les longueurs $\boldsymbol{w}$.

$$\begin{cases} \boldsymbol{w} & = & \underset{\boldsymbol{v} \in \mathbb{R}^{N_w}}{\mathrm{argmax}}\ \ln(L(\boldsymbol{v})) \\ \ln(L(\boldsymbol{w})) & = & -0.5 \left(N_{DOE} \left(\ln(\sigma_K^2(\boldsymbol{w})) + \ln(2\pi) + 1\right) + \ln\left(|\boldsymbol{\Gamma}(\boldsymbol{w})|\right)\right) \end{cases}$$

Ceci revient à minimiser une fonction, comme détaillé à l'équation (3.2), qui s'écrit donc ici :

$$\mathcal{E}\left(\hat{f}\left(\bullet;\ \boldsymbol{w}\right), DOE\right) = 0.5\left(N_{DOE}\left(\ln(\sigma_K^2(\boldsymbol{w})) + \ln(2\pi) + 1\right) + \ln\left(|\boldsymbol{\Gamma}(\boldsymbol{w})|\right)\right) \tag{3.23}$$

Ce problème d'optimisation peut être compliqué à résoudre. En effet, la fonction du maximum de vraisemblance est caractérisée par des gradients quasi-nuls dans certaines zones et par la présence de nombreux optima locaux [Watkins, 1992]. Certaines études préconisent l'utilisation d'optimiseurs sans gradient tels que des algorithmes évolutionnaires du type évolution différentielle [Zhang *et al.*, 2010], d'autres utilisent des algorithmes locaux avec ou sans gradient avec plusieurs initialisations (c'est notamment ce qui est implémenté dans scikit-learn [Pedregosa *et al.*, 2011]).

Par ailleurs, Bachoc a montré dans [Bachoc, 2013], que le choix des paramètres par maximum de vraisemblance peut commettre une erreur importante dans l'estimation des paramètres. Plus précisément, si la forme analytique du noyau $\boldsymbol{R}$ est mal choisie, la technique par validation croisée (cf algorithme 3.2) du type *leave-one-out* permet d'obtenir des modèles de krigeage commettant une erreur de prédiction moins importante [Rasmussen et Williams, 2005].

3.2.3.3 Quelques remarques :

Choix de R et effet sur la prédiction $\hat{f}$: comme cela a déjà été mentionné précédemment, la régularité du noyau donne la régularité de la prédiction de $\hat{f}$ puisque, dans le cas du krigeage ordinaire, on peut réécrire l'équation (3.19) de la manière suivante :

$$\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) = a_0 + \sum_{k=1}^{N_{DOE}} a_k R\left(\boldsymbol{x}, \boldsymbol{x}^k;\ \boldsymbol{w}\right)$$

avec $(\boldsymbol{x}^k)_{1\leqslant k\leqslant N_{DOE}}$ les points du DOE et les $(a_k)_{1\leqslant k\leqslant N_{DOE}}$ indépendants de la variable $\boldsymbol{x}$. On voit donc que c'est une combinaison linéaire de ces noyaux de corrélation, donc conservant la même forme que ces derniers. Ceci explique pourquoi il est très important de choisir ces noyaux de manière adaptée. Il en existe d'autres que les noyaux gaussiens, par exemple la famille des noyaux de Matérn permettant d'avoir une régularité donnée, des noyaux linéaires ou cubiques si la fonction à prédire est polynomiale, voire des noyaux avec des exponentielles de valeur absolue si la fonction n'est pas différentiable :

$$R(\boldsymbol{x}, \boldsymbol{x}';\ \boldsymbol{w}) = \exp\left(-\sum_{i=1}^{N_x} \frac{|x_i - x_i'|}{w_i}\right)$$

Par ailleurs, pour insister sur l'importance de ce noyau de corrélation, Bachoc montre dans [Bachoc, 2013] que l'erreur de prédiction d'un modèle de krigeage est très sensible au choix de ce noyau.

Linéarité de la prédiction $\hat{f}$ par rapport aux observations $\boldsymbol{Y}$: en réécrivant l'équation (3.19) mais cette fois-ci en factorisant par les observations $\boldsymbol{Y}$, on peut écrire :

$$\hat{f}(\boldsymbol{x};\ \boldsymbol{w}) = a_0'(\boldsymbol{x}) + \sum_{k=1}^{N_{DOE}} a_k'(\boldsymbol{x}) y^k$$

avec $(a'_k(\boldsymbol{x})_{1\leqslant k\leqslant N_{DOE}}$ indépendants des observations $\boldsymbol{Y}$. Ce qui montre que la prédiction est en réalité une combinaison linéaire des sorties des points du DOE. Comme la remarque précédente le montre, ceci ne signifie en revanche pas que le modèle est linéaire par rapport à $\boldsymbol{x}$.

À paramètres du krigeage fixés, l'estimation de l'erreur ne dépend pas des observations $\boldsymbol{Y}$. En effet, en regardant l'équation (3.22), il apparaît que l'erreur d'estimation est ici totalement indépendante des valeurs prises par la fonction à approcher f. Ceci peut avoir un intérêt lorsque l'on souhaite connaître l'effet du rajout d'un point sur l'incertitude du modèle, notamment lors de la construction séquentielle d'un DOE. Ce point est détaillé dans la section 3.2.4.

Cas où f est à plusieurs sorties : le cas où la fonction à prédire f a plusieurs sorties peut être traité en séparant chacune de ces sorties et en les traitant indépendamment. Le cas où l'on considère que les sorties ne sont pas indépendantes s'appelle le cokrigeage. Cette technique peut notamment être utilisée afin de prendre en compte le gradient de la fonction en plus de sa valeur en chaque point du DOE [Ulaganathan *et al.*, 2015] [Han *et al.*, 2013].

Utilité des techniques de sélection de modèle : Du fait que la complexité C d'un modèle de krigeage est fixée une fois le noyau R choisi, les techniques de sélection de modèle présentées en section 3.1.2 ne sont pas nécessairement utiles.

Mais des techniques s'apparentant à la *ridge regression* peuvent toutefois être intéressantes dans le cas du krigeage. En effet, pour calculer la prédiction par krigeage et la variance de cette estimation, il est nécessaire d'inverser la matrice de corrélation $\boldsymbol{\Gamma}$ (voir les équations (3.19) et (3.22)). Or, cette matrice peut devenir très mal conditionnée, par exemple dans le cas où deux points du DOE sont très proches l'un de l'autre. Une manière permettant d'améliorer ce conditionnement peut être d'utiliser un effet pépite. Ceci consiste à rendre le modèle de krigeage régressant en introduisant un terme constant sur la diagonale de la matrice $\boldsymbol{\Gamma}$, σ_N^2. Cette amélioration du conditionnement peut aussi rendre l'apprentissage des longueurs de corrélation plus facile. Le calcul des $\boldsymbol{w}$ est détaillé dans la section suivante.

Ce terme constant correspond en fait à la variance d'un bruit virtuel sur la fonction à prédire. Le choix de ce paramètre peut donc poser problème lorsque la fonction à prédire n'est pas bruitée : il est en effet important de ne pas choisir un σ_N trop important, de sorte à ne pas trop s'éloigner de la vraie fonction. L'étude de Peng [Peng et Wu, 2014] fait le lien avec la *ridge regression* et détaille les conséquences de l'utilisation de l'effet pépite dans le cas où la fonction à prédire est déterministe. Enfin, il propose une méthode d'estimation basée sur la validation croisée. Ce paramètre peut également être estimé par maximum de vraisemblance.

3.2.3.4 Mise en œuvre du krigeage

Dans le cadre de cette étude, le krigeage a été mis en place à l'aide du *package* Scikit-learn codé en Python [Pedregosa *et al.*, 2011]. Il permet de construire des modèles de krigeage avec des noyaux gaussiens ou exponentiels, avec effet pépite ou non, anisotropes ou non.

L'optimisation des paramètres se fait principalement avec l'algorithme local sans gradient COBYLA introduite par Powell [Powell, 1998].

3.2.4 Construction séquentielle de plan d'expériences

La limite des méthodes *a priori* présentées en section 3.1.3 est que, par définition, elles n'utilisent à aucun moment les caractéristiques de la fonction à prédire f. Or, cela peut avoir pour inconvénient de mailler trop finement des zones dont la connaissance n'apporte pas d'information supplémentaire au modèle de substitution.

L'idée est donc de partir d'un plan initial de taille réduite DOE^0 permettant de construire un premier modèle de substitution $\hat{f}$, puis de l'enrichir en utilisant l'estimateur d'erreur de ce $\hat{f}$. La génération de DOE^0 peut quant à elle être réalisée par LHS. Les étapes de génération d'un tel DOE sont résumées dans l'algorithme 3.5. Ces stratégies sont particulièrement répandues dans le cas du krigeage puisque son estimateur d'erreur est connu analytiquement comme le montre l'équation (3.22). En dehors du cas du krigeage, l'estimation de ces erreurs est possible par la méthode du *bootstrap* (voir l'équation (3.7) pour sa mise en place) mais représente un coût d'estimation important. Il est donc plus rare de le rencontrer.

Algorithme 3.5 Construction d'un DOE par enrichissement séquentiel

Input : La taille du DOE initial N^0_{DOE}, le budget total pour le DOE N_{DOE}, le critère d'enrichissement $\mathcal{I}$nfo
Output : DOE

· Générer les N^0_{DOE} entrées du DOE initial $\boldsymbol{X}^0$ par LHS;
· Évaluer $\boldsymbol{Y}^0 = f(\boldsymbol{X}^0) \rightarrow DOE^0$;
· Soit $N_{courant} := N^0_{DOE}$ et $DOE^{courant} := DOE^0$;
while $N_{courant} < N_{DOE}$ **do**
  · Construire $\hat{f}$ avec $DOE^{courant}$;
  · Choisir $\boldsymbol{x}^\star = \text{argmax}_{\boldsymbol{x}\in\mathcal{D}_x}\ \mathcal{I}\text{nfo}(\boldsymbol{w}, \boldsymbol{x})$;
  · $DOE^{courant} = \{DOE^{courant}, (\boldsymbol{x}^\star, f(\boldsymbol{x}^\star))\}$;
end
$DOE := DOE^{courant}$;

Le point à ajouter au DOE à chaque itération, $\boldsymbol{x}^\star$, est celui qui maximise la quantité d'information qu'apporte un point $\boldsymbol{x}$ au modèle. À cette fin, il est nécessaire de définir une fonction d'information $\mathcal{I}\text{nfo} : \mathbb{R}^{N_w} \times \mathbb{R}^{N_x} \longrightarrow \mathbb{R}$. Dans un souci de simplicité, cette fonction est uniquement écrite avec sa dépendance en $\boldsymbol{x}$: $\mathcal{I}\text{nfo}(\boldsymbol{x})$.

Selon ce que l'utilisateur veut faire de son modèle, la formulation de cette information varie. Par exemple, si le modèle de substitution est utilisé en optimisation ou pour estimer des mesures probabilistes telles que le quantile, il n'est pas intéressant de connaître parfaitement le modèle sur tout le domaine $\mathcal{D}_x$ et seules certaines zones à cibler doivent être enrichies. Dans cette partie, l'objectif est de construire un modèle qui commet l'erreur globale la plus faible possible. À cette fin, on peut citer deux principaux critères d'enrichissement séquentiels :

1. Le premier est le plus intuitif et le plus simple à mettre en œuvre et consiste à choisir le point qui maximise la variance du modèle, c'est-à-dire :

$$\mathcal{I}\text{nfo}(\boldsymbol{x}) = \hat{\sigma}(\boldsymbol{x}) \tag{3.24}$$

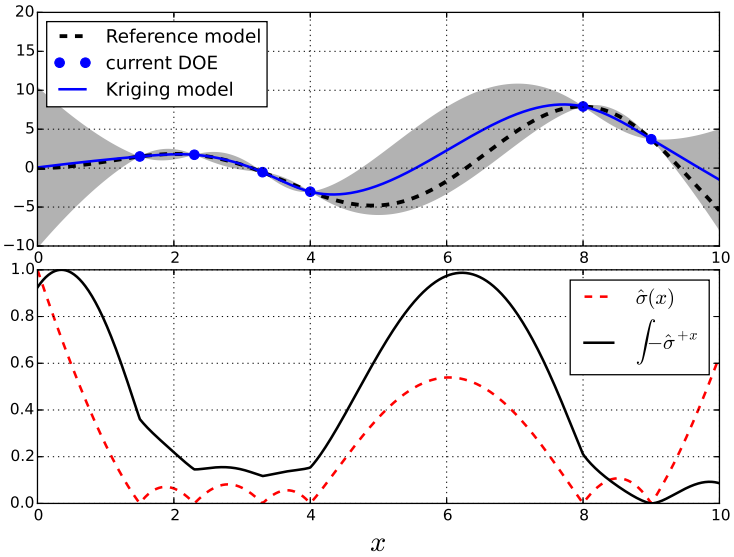


FIGURE 3.13 – Illustration des deux critères d'enrichissement possibles rendus sans dimension entre 0 et 1.

Le principal inconvénient de ce critère est qu'il enrichit les bords du domaine où l'erreur de krigeage est souvent la plus importante, sans forcément apporter plus d'information sur la fonction elle-même. Ceci est illustré sur la figure 3.13 avec la courbe en pointillés rouges. En effet, on voit que les deux points qu'il faudrait ajouter dans ce cas sont les bords, alors que la fonction n'y fait pas une erreur importante par rapport à la zone $4 \leqslant x \leqslant 8$. Ce problème se pose d'autant plus pour une dimension de $\boldsymbol{x}$ supérieure puisque le nombre de bords grandit exponentiellement avec la dimension.

2. Le second permet de résoudre ce problème. Pour cela, son principe repose sur le calcul de l'effet du rajout d'un point sur l'erreur moyenne. Ce critère s'appelle IMSE pour *Integrated Mean Squared Error* [Santner *et al.*, 2003] Soit $\hat{\sigma}^{+x}$ l'estimation de l'erreur après l'ajout fictif d'un point dans le DOE donc : $\boldsymbol{X}^{+x} = \{\boldsymbol{X}, \boldsymbol{x}\}$. Sans avoir besoin de calculer le vrai modèle en ce point, il est possible d'avoir une estimation de la variance fictive du modèle $\hat{\sigma}^{+x}$ après ajout et ce quel que soit le type de modèle utilisé. L'intégration de cette variance fictive sur tout le domaine fournit une mesure de l'effet de l'ajout du point $\boldsymbol{x}$ sur l'erreur commise par $\hat{f}$ sur tout le domaine, donc :

$$\mathcal{I}\text{nfo}(\boldsymbol{x}) = -\int_{\boldsymbol{u}\in\mathcal{D}_x} \hat{\sigma}^{+\boldsymbol{x}}(\boldsymbol{u}) \tag{3.25}$$

Le signe $-$ se justifie par le fait que l'information apportée est maximale lorsque l'erreur globale, estimée par cette intégrale, est la plus faible. Cette intégrale peut par exemple être estimée par des méthodes de Monte-Carlo. Sur la figure 3.13, la courbe en noir représente dans le cas du krigeage cette mesure d'information. Cette fois-ci, les bords ne sont plus les points où se réalise le maximum. Notamment, la zone $4 \leqslant x \leqslant 8$ gagne logiquement en importance avec ce nouveau critère.

Le détail de l'estimation du premier critère a déjà été réalisé. En revanche, l'estimation de la variance fictive $\hat{\sigma}^{+x}$ de la prédiction du modèle de substitution après l'ajout d'un

point $\boldsymbol{x}$ n'est pas immédiate et doit être expliquée.

3.2.4.1 Estimation de la variance de la prédiction du modèle après l'ajout fictif d'un point au DOE

Cas du krigeage Comme cela a été détaillé en section 3.2.3.3, l'estimation de l'erreur du krigeage (voir l'équation (3.22)) ne dépend pas de la valeur de la sortie de la fonction à prédire f. De cette manière, en conservant les longueurs de corrélation $\boldsymbol{w}$ et le σ_K calculés avec les valeurs provenant de la fonction à prédire f, il est possible de reconstruire $\boldsymbol{\Gamma}^{+\boldsymbol{x}^{new}}$ et $\gamma^{+\boldsymbol{x}^{new}}$ afin de quantifier l'effet de l'ajout d'un point $\boldsymbol{x}^{new}$ sur l'estimation de $\hat{\sigma}$:

$$\gamma^{+\boldsymbol{x}^{new}}(\boldsymbol{x}) = \Big(R(\boldsymbol{x}, \boldsymbol{x}^1;\ \boldsymbol{w}), \cdots, R(\boldsymbol{x}, \boldsymbol{x}^{N_{DOE}};\ \boldsymbol{w}), R(\boldsymbol{x}, \boldsymbol{x}^{new};\ \boldsymbol{w})\Big)$$

$$\boldsymbol{\Gamma}^{\boldsymbol{x}^{new}} = \left(\begin{array}{c|c} \multirow{2}{*}{$\boldsymbol{\Gamma}$} & R\left(\boldsymbol{x}^1, \boldsymbol{x}^{new};\ \boldsymbol{w}\right) \\ & \vdots \\ \hline R\left(\boldsymbol{x}^{new}, \boldsymbol{x}^1;\ \boldsymbol{w}\right) \quad \cdots \quad R\left(\boldsymbol{x}^{new}, \boldsymbol{x}^{N_{DOE}};\ \boldsymbol{w}\right) & R\left(\boldsymbol{x}^{N_{DOE}}, \boldsymbol{x}^{new};\ \boldsymbol{w}\right) \\ & R\left(\boldsymbol{x}^{new}, \boldsymbol{x}^{new};\ \boldsymbol{w}\right) \end{array}\right)$$

Enfin, l'erreur de prédiction s'estime de la même manière qu'à l'équation (3.22), mais en remplaçant γ par $\gamma^{+\boldsymbol{x}^{new}}$ et $\boldsymbol{\Gamma}$ par $\boldsymbol{\Gamma}^{\boldsymbol{x}^{new}}$:

$$\hat{\sigma}^{+\boldsymbol{x}^{new}}(\boldsymbol{x})^2 = \sigma_K^2 \left(1 - \gamma^{+\boldsymbol{x}^{new}}(\boldsymbol{x})^T \boldsymbol{\Gamma}^{+\boldsymbol{x}^{new}\,-1} \gamma^{+\boldsymbol{x}^{new}}(\boldsymbol{x}) + \frac{1 - \mathbf{1}^T \boldsymbol{\Gamma}^{+\boldsymbol{x}^{new}\,-1} \gamma^{+\boldsymbol{x}^{new}}(\boldsymbol{x})}{\mathbf{1}^T \boldsymbol{\Gamma}^{+\boldsymbol{x}^{new}\,-1} \mathbf{1}}\right) \tag{3.26}$$

Estimation de l'erreur pour les autres modèles Dans le cas d'un modèle ne fournissant pas intrinsèquement une estimation de l'erreur, il est possible d'utiliser des techniques de *bootstrap* comme expliqué à l'algorithme 3.7. Cependant, à la différence de l'estimation de la variance par krigeage, celle fournie par *bootstrap* nécessite de connaître la valeur de la sortie $f(\boldsymbol{x}^{new})$ en le point à rajouter $\boldsymbol{x}^{new}$. Afin d'appliquer classiquement le *bootstrap*, il est nécessaire d'avoir un DOE virtuel $DOE^{+\boldsymbol{x}^{new}} = \{DOE, (\boldsymbol{x}^{new}, \boldsymbol{y}^{new})\}$ sur lequel l'appliquer et donc de faire une hypothèse sur la valeur de $\boldsymbol{y}^{new}$. Par exemple, il est possible de faire confiance au modèle de substitution et prendre $\boldsymbol{y}^{new} = \hat{f}(\boldsymbol{x}^{new};\ \boldsymbol{w})$.

Toutefois, le calcul de la variance $\hat{\sigma}^{+\boldsymbol{x}^{new}}(\boldsymbol{x})$ à partir de la formule 3.7 implique la construction de B modèles de substitution pour chaque $\boldsymbol{x}^{new}$. À cause de cela, il peut être inenvisageable de maximiser l'intégrale de ce critère qu'il faut estimer pour évaluer $\mathcal{I}\text{nfo}$ à l'aide de l'équation (3.25). L'utilisation de ce critère dépend donc du temps de calcul disponible et du coût de construction d'un modèle, sachant que chaque estimation de $\mathcal{I}\text{nfo}(\boldsymbol{x})$ demande la construction de B modèles de substitution.

3.3 Construction de modèles de substitution de codes spatio-temporels

La section précédente a détaillé la construction de modèles de substitution pour un modèle stationnaire, c'est-à-dire dans le cas où les entrées $\boldsymbol{x}$ et les sorties $\boldsymbol{y}$ de ce modèle sont indépendantes du temps. Toutefois, il existe également le besoin en ingénierie d'approcher des modèles dont les entrées et les sorties peuvent dépendre du temps. Par exemple, lors de la résolution de systèmes d'équations différentielles, ou plus généralement lors de l'utilisation de simulations numériques d'un système dynamique, les entrées et sorties dépendent du temps, ce qui nécessite des modifications aux techniques présentées précédemment. Plus précisément, il est donc nécessaire de prédire des sorties du type $\boldsymbol{y}(t) = (y_1(t), \cdots, y_{N_y}(t))$ à l'aide d'entrées exogènes du type $\boldsymbol{u}(t) = (u_1(t), \cdots, u_{N_u}(t))$. Ces entrées exogènes peuvent par exemple être la traduction des conditions aux limites et termes forçants de simulations numériques instationnaires.

Du fait que les entrées et sorties peuvent provenir de capteurs, numériques ou expérimentaux, les différentes fonctions $\boldsymbol{y}(t)$ et $\boldsymbol{u}(t)$ ne sont donc connues qu'à certains instants $t_0 < t_1 < \cdots < t_k \leqslant T$. Dès lors, l'objectif est plus spécifiquement de prédire l'évolution des sorties $\boldsymbol{y}^k := \boldsymbol{y}(t^k)\ \forall k \geqslant 0$ uniquement à partir de la connaissance de la valeur initiale de ces sorties : $\boldsymbol{y}^0 := \boldsymbol{y}(t^0)$ et des entrées exogènes jusqu'au temps t^k : $\boldsymbol{u}^0 := \boldsymbol{u}(t^0)$, $\cdots$, $\boldsymbol{u}^k := \boldsymbol{u}(t^k)$. Les valeurs prédites par le modèle de substitution sont écrites $\hat{\boldsymbol{y}}^k$ afin de les différencier des valeurs mesurées, c'est-à-dire provenant du système à prédire, $\boldsymbol{y}^k$.

Les méthodes les plus répandues pour la construction de modèles de substitution dynamiques de ce type sont celles issues de l'automatique et basées sur l'identification de la structure récurrente ou non à utiliser. Une fois cette structure identifiée, la forme de la boîte noire peut par exemple être un réseau de neurones rebouclé à une couche cachée. Les caractéristiques de ces réseaux ainsi que les techniques d'apprentissage sont présentés à la suite. Enfin, il existe une autre modélisation dynamique possible basée sur les processus gaussiens et détaillée dans la dernière partie.

3.3.1 Identification de système pour la modélisation dynamique de boîtes noires

La théorie de l'identification a notamment été développée par Ljung [Ljung, 1999] puis reprise et adaptée au cas des réseaux de neurones par Dreyfus [Dreyfus *et al.*, 2002]. L'identification de système est une branche de l'automatique consistant à modéliser un système dynamique uniquement à partir d'observations de ce dernier. Afin d'identifier le modèle le plus adapté, l'utilisateur doit faire des choix de construction basés sur les *a priori* disponibles sur ce processus. Ces *a priori* permettent de : 1) générer le DOE spatio-temporel 2) de déterminer l'ordre en temps (qui est un retard appliqué sur les entrées) 3) de déterminer le modèle-hypothèse puis sa mise en œuvre par simulation ou prédiction (cf figure 3.14). Le modèle-hypothèse est plus précisément la traduction de l'*a priori* que possède l'utilisateur sur le processus à modéliser, c'est-à-dire sur la relation déterministe reliant les entrées exogènes aux sorties choisies. Les *a priori* à prendre en compte peuvent être [Oussar, 1998] :

— **La linéarité ou non** de la physique à modéliser
— **La présence d'un bruit ou non** dans le processus. Si l'on considère que le système est bruité, ces bruits peuvent être de deux types : intrinsèques au processus, dans

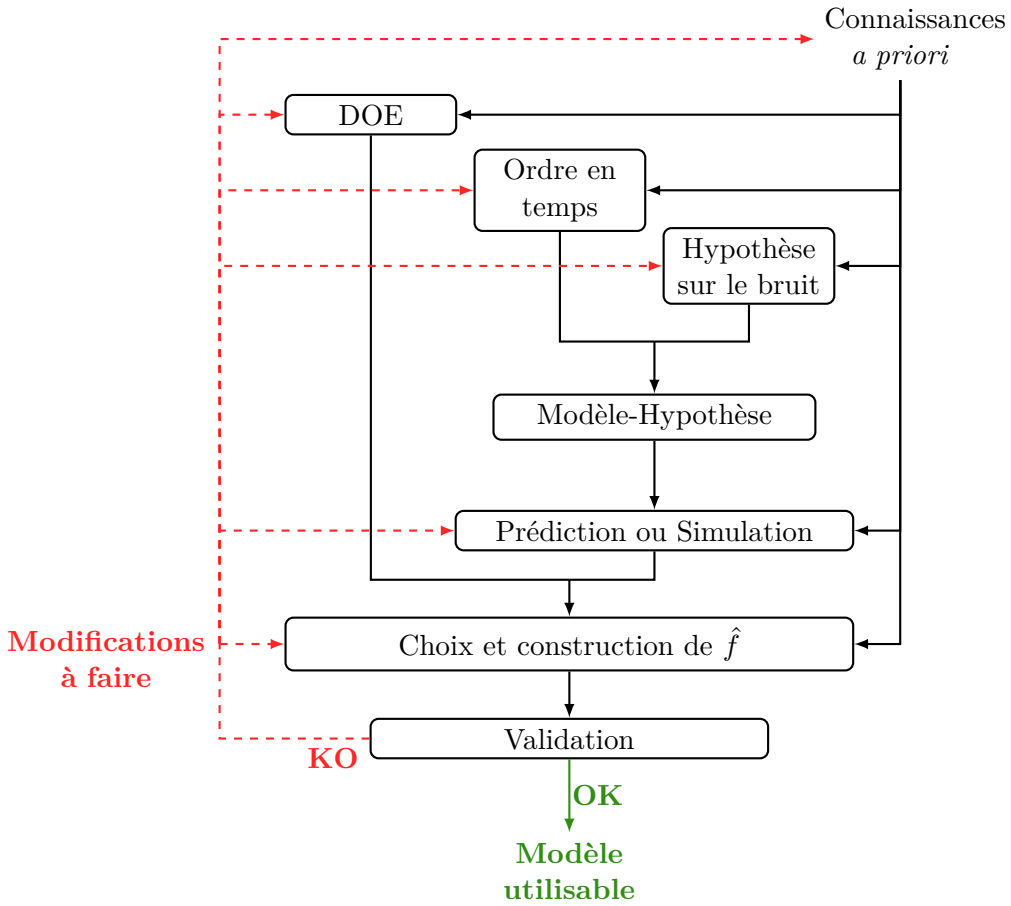


FIGURE 3.14 – Schéma explicatif de l'identification de système

ce cas il est appelé bruit d'état ou provenant de la mesure de la sortie du processus appelé bruit de sortie.

— **L'ordre temporel** du modèle à prédire : par exemple l'équation de la chaleur (cf équation (2.1)) est une équation du premier ordre en temps alors que l'équation d'onde est une équation d'ordre 2 en temps. Cet *a priori* permet notamment de faire des modèles "boîte grise", c'est-à-dire utilisant l'information provenant du système d'équations aux dérivées partielles régissant le processus.

— **La forme temporelle des entrées exogènes à prédire**. Cette information peut aider à générer le DOE spatio-temporel en fournissant une base de fonction sur laquelle générer ce DOE. Par exemple, si les trajectoires à prédire sont de forme sinusoïdale, il peut être intéressant de choisir une base sinusoïdale pour générer les trajectoires d'apprentissage.

3.3.1.1 Choix du modèle-hypothèse pour traiter le bruit

Il est possible de classifier les modèles-hypothèses selon l'hypothèse faite sur le bruit. En effet, selon l'aspect déterministe ou bruité du système lui-même (bruit d'état) ou des mesures provenant du système (bruit de sortie), les modèles-hypothèses recommandés ne sont pas les mêmes. On peut donc les classer de quatre sortes : les modèles-hypothèses

déterministes, les modèles-hypothèses à bruit de sortie, les modèles-hypothèses à bruit d'état et enfin les modèles-hypothèses à bruit de sortie et d'état :

1. Modèle-hypothèse déterministe : On considère que le processus est parfait, c'est-à-dire sans bruit. Ce qui nous donne une relation entrée-sortie hypothèse pour le processus à prédire f suivant :

$$\boldsymbol{y}^k = f\left(\boldsymbol{u}^{k-q}, \cdots, \boldsymbol{u}^{k-1}, \boldsymbol{y}^{k-q}, \cdots, \boldsymbol{y}^{k-1}\right) \tag{3.27}$$

   avec q l'ordre temporel du système.

2. Modèle-hypothèse à bruit de sortie : ou *output error* en anglais. Soit ε^k le bruit, qui est une variable aléatoire de moyenne nulle. Un bruit de sortie correspond à un bruit sur la mesure $\boldsymbol{y}^k$ de l'état du système $\boldsymbol{z}^k$ à chaque temps de mesure t^k. Ce qui se traduit par l'équation suivante :

$$\left\{ \begin{array}{rcl} \boldsymbol{z}^k & = & f\left(\boldsymbol{u}^{k-q}, \cdots, \boldsymbol{u}^{k-1}, \boldsymbol{z}^{k-q}, \cdots, \boldsymbol{z}^{k-1}\right) \\ \boldsymbol{y}^k & = & \boldsymbol{z}^k + \varepsilon^k \end{array} \right. \tag{3.28}$$

   Plus précisément, l'état réel du système $\boldsymbol{z}^k$ est inconnu et ne peut être fourni à l'utilisateur qu'à travers la mesure que l'on peut en faire, donc $\boldsymbol{y}^k$.

3. Modèle-hypothèse à bruit d'état : appelé également *equation error* ou NARX (pour *Non-linéaire Auto-Régressif à entrées eXogènes*) dans le domaine de la modélisation non-linéaire. Soit ε^k le bruit, qui est une variable aléatoire de moyenne nulle. Le bruit d'état est un bruit intrinsèque au modèle et s'écrit donc, lors de la prédiction à chaque itération de la manière suivante :

$$\boldsymbol{y}^k = f\left(\boldsymbol{u}^{k-q}, \cdots, \boldsymbol{u}^{k-1}, \boldsymbol{y}^{k-q}, \cdots, \boldsymbol{y}^{k-1}\right) + \varepsilon^k \tag{3.29}$$

   Cette fois-ci, le bruit fait donc partie du système et n'est plus une erreur due à sa mesure.

4. Modèle-hypothèse à bruit d'état et de sortie : aussi appelé NARMAX pour *Non-linéaire Auto-Régressif à Moyenne Ajustée et entrées eXogènes* dans le domaine de la modélisation non-linéaire. Ce n'est en fait que le mélange des deux équations précédentes (3.29) et (3.28) avec deux bruits indépendants ε_e^k et ε_s^k présents dans le système :

$$\left\{ \begin{array}{rcl} \boldsymbol{z}^k & = & f\left(\boldsymbol{u}^{k-q}, \cdots, \boldsymbol{u}^{k-1}, \boldsymbol{z}^{k-q}, \cdots, \boldsymbol{z}^{k-1}\right) + \varepsilon_e^k \\ \boldsymbol{y}^k & = & \boldsymbol{z}^k + \varepsilon_s^k \end{array} \right. \tag{3.30}$$

3.3.1.2 Prédicteur ou simulateur pour la mise en pratique de ces modèles-hypothèses

Une fois le modèle-hypothèse choisi, il faut lui associer une manière d'utiliser le modèle de substitution $\hat{\boldsymbol{f}}$. Il existe deux utilisations possibles : par prédicteur ou par simulateur.

— Les prédicteurs sont des modèles fonctionnant en parallèle avec le système : c'est-à-dire qu'ils permettent d'estimer la valeur de la sortie au temps t^k, $\hat{\boldsymbol{y}}^k$, à l'aide des entrées exogènes et des sorties **mesurées** aux q temps précédents et de l'état

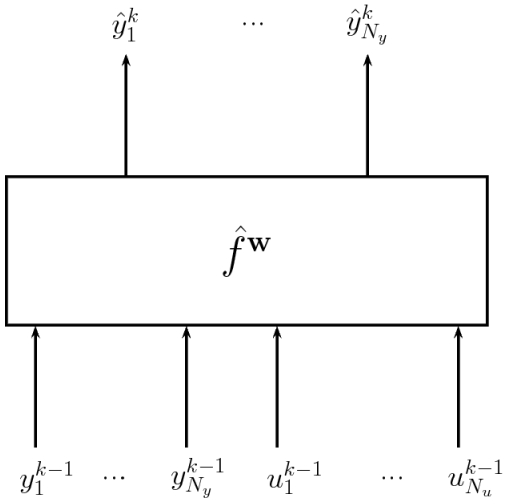


FIGURE 3.15 – Illustration d'un prédicteur à un pas

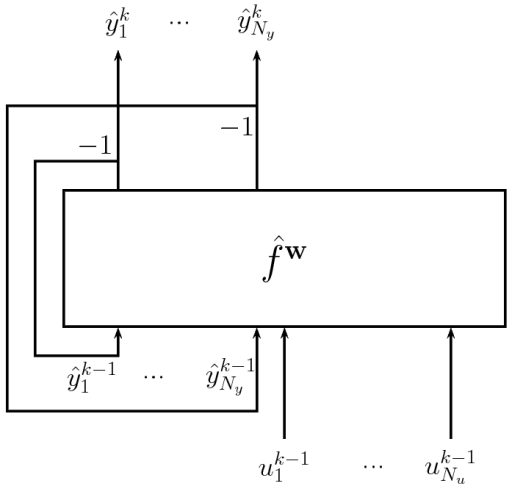


FIGURE 3.16 – Illustration d'un simulateur à un pas

initial $\boldsymbol{y}^0$. Par exemple dans le cas déterministe sans bruit, cela revient à utiliser $\hat{f}$ de la manière suivante :

$$\hat{\boldsymbol{y}}^k = \hat{f}\left(\boldsymbol{u}^{k-q}, \cdots, \boldsymbol{u}^{k-1}, \boldsymbol{y}^{k-q}, \cdots, \boldsymbol{y}^{k-1};\ \boldsymbol{w}\right) \tag{3.31}$$

Le cas $q = 1$ est appelé prédicteur à un pas et est illustré sur la figure 3.15. Ces modèles ont pour avantage d'être faciles à mettre en œuvre puisqu'ils ne nécessitent pas l'utilisation de réseaux de neurones rebouclés. Grâce à cela, l'apprentissage de ces modèles peut se faire de manière tout à fait classique, avec des entrées mesurées pour chaque valeur de k : on appelle cela l'apprentissage dirigé, ou *teacher forcing* en anglais.
Toutefois, les prédicteurs nécessitent de fonctionner simultanément avec le modèle réel de sorte à connaître la valeur des sorties mesurées pour chaque prédiction. Ils ne sont donc pas adaptés au cadre de cette étude.

— Les simulateurs sont des modèles indépendants du système à substituer. Ceci signifie que pour prédire la sortie au temps t^k, $\hat{\boldsymbol{y}}^k$, ils ne nécessitent que la connaissance des entrées exogènes aux q temps précédents. En effet, les sorties au temps précédent utilisées par le simulateur sont celles qu'il a prédites lui-même. Par exemple dans le cas sans bruit, cela revient à utiliser la formule suivante :

$$\hat{\boldsymbol{y}}^k = \hat{f}\left(\boldsymbol{u}^{k-q}, \cdots, \boldsymbol{u}^{k-1}, \hat{\boldsymbol{y}}^{k-q}, \cdots, \hat{\boldsymbol{y}}^{k-1};\ \boldsymbol{w}\right) \tag{3.32}$$

Ce qui correspond au cas d'un réseau de neurone bouclé cf figure 3.16. Ces modèles pouvant entièrement se substituer au système d'origine, ils sont donc très adaptés pour faire de la prédiction dans le cas où les simulations numériques fournissant les sorties mesurées sont coûteuses. En revanche, leur apprentissage est plus complexe puisqu'en dehors du premier pas de temps, ils utilisent en entrée des sorties rebouclées, donc elles-mêmes prédites par $\hat{f}$. Le modèle est donc partiellement autonome et ce dès l'apprentissage, c'est-à-dire qu'il n'a que les valeurs des entrées exogènes qui sont mesurées. Ce type d'apprentissage est donc appelé apprentissage semi-dirigé.

En reprenant la classification par type de bruit, on remarque que chacun de ces modèles-hypothèses peut être optimalement traité par une utilisation de type simulateur et/ou de type prédicteur. C'est-à-dire que si l'utilisation en forme prédicteur ou simulateur d'un modèle $\hat{f}$ supposé identique à f ne commet pas plus d'erreur que le bruit lui-même, alors il est considéré comme modèle optimal.

Par exemple dans le cas d'un bruit d'état, il peut être montré [Dreyfus *et al.*, 2002] que l'utilisation en prédicteur 3.31 à un pas est la manière commettant le moins d'erreur dans l'approximation du processus. En effet, si après apprentissage, la fonction $\hat{f}$ est identique à f, alors l'erreur de prédiction commise par la séquence $\hat{\boldsymbol{y}}^k$ prédite par 3.31 pour estimer le $\boldsymbol{y}^k$ de l'équation (3.28) est donc égale à $\boldsymbol{y}^k - \hat{\boldsymbol{y}}^k = \varepsilon^k$, ce qui est exactement égal au bruit. Ceci signifie que c'est le modèle idéal puisqu'il ne modélise que la partie déterministe sans prendre en compte le bruit.

De la même manière, Dreyfus classifie les différents modèles-hypothèses selon leur utilisation optimale. Les conclusions sont présentées dans le tableau 3.1. On y remarque notamment que la présence d'un bruit d'état, combiné à un bruit de sortie ou non, nécessite l'utilisation de méthode à un pas étant donné qu'il est nécessaire de corriger l'erreur provenant de ce bruit d'état à chaque itération. Si l'hypothèse de la présence de bruit d'état est vraie et que les méthodes mises en place ne le prennent pas en compte, cela peut induire une propagation de ce bruit dans le simulateur, menant à des instabilités dans les prédictions.

Enfin, étant donné que la résolution numérique d'un système d'équations différentielles est considérée parfaite donc sans bruit, les deux utilisations (3.31) ou (3.29) peuvent être adaptées, selon ce que souhaite faire l'utilisateur. Tous ces résultats sont résumés dans le tableau suivant :

Hypothèse sur le bruit	Nom du modèle non-linéaire	Utilisation	Apprentissage
Bruit d'état	NARX	Prédicteur à un pas	Dirigé
Bruit de sortie	*Output error*	Simulateur	Semi-dirigé
Bruit d'état et de sortie	NARMAX	Prédicteur à un pas	Semi-dirigé
Pas de bruit		Prédicteur ou Simulateur	Dirigé ou Semi-dirigé

TABLEAU 3.1 – Résumé des modèles-hypothèses avec les utilisations et apprentissages conseillés

3.3.1.3 Choix de l'ordre en temps à partir d'un système d'équations différentielles - Modèle boîte grise

Lors de la réduction d'un modèle spatio-temporel, il est possible que l'utilisateur connaisse le système d'équations différentielles à l'origine des observations de $\boldsymbol{y}^k$. Cela peut en particulier être le cas de données provenant de simulations numériques coûteuses, que l'on souhaite substituer. Ce système d'équations peut permettre de déterminer l'ordre en temps du modèle-hypothèse, la variable q dans l'équation (3.32).

Par exemple, Le Riche, dans son étude [Riche *et al.*, 2001], utilise cette idée pour déduire les charges subies par un moyeu de roue de voiture à partir des accélérations mesurées de ce dernier au cours du temps. Autre exemple en ingénierie thermique, De Lozzo dans [Lozzo *et al.*, 2013] part de l'équation de la chaleur (écrite à l'équation (2.1)) pour justifier l'utilisation d'un simulateur (donné à l'équation (3.32)) à un pas basé sur les réseaux de neurones récurrents. En effet, en discrétisant ce système spatialement avec des nœuds comme pour le modèle nodal présenté (cf équation (2.7) en section 2.2.1.1) et temporellement à l'aide d'un schéma explicite en temps, on obtient un système d'équation de la forme suivante :

$$\left\{\begin{array}{rcl} y_1^k & = & f_1(u_1^{k-1},\cdots,u_{N_u}^{k-1},y_1^{k-1},\cdots,y_{N_y}^{k-1}) \\ & \cdots & \\ y_j^k & = & f_j(u_1^{k-1},\cdots,u_{N_u}^{k-1},y_1^{k-1},\cdots,y_{N_y}^{k-1}) \\ & \cdots & \\ y_{N_y}^k & = & f_{N_y}(u_1^{k-1},\cdots,u_{N_u}^{k-1},y_1^{k-1},\cdots,y_{N_y}^{k-1}) \end{array}\right. \tag{3.33}$$

avec y_j^k représentant la température au temps t^k et au nœud j et les u_i^{k-1} représentant les conditions aux limites et termes forçants connus et dépendant du temps. Par exemple, ces entrées exogènes peuvent être la température ambiante ou la puissance dissipée dans l'équipement électronique.

Du fait de la discrétisation explicite en temps, il est toutefois nécessaire de faire attention à la stabilité du système : la discrétisation temporelle des données d'entrée ne doit pas être trop grande. Si les points des simulations numériques d'origine viennent d'une résolution basée sur un schéma temporel explicite, cette contrainte est automatiquement vérifiée. Si ce n'est pas le cas, il est important de faire attention à la stabilité lors de l'apprentissage et l'utilisation de $\hat{f}$.

3.3.2 Réseaux de neurones récurrents

Les réseaux de neurones récurrents sont un choix très apprécié pour la forme analytique du simulateur $\hat{f}$. Ils ont l'avantage d'être des estimateurs parcimonieux et adaptés aux systèmes non-linéaires comme expliqué en section 3.2.2. De plus, même avec une utilisation rebouclée, il est prouvé qu'ils conservent leur propriété d'approximateur universel [Sontag, 1997].

Les premiers réseaux rebouclés utilisés pour la prédiction existant dans la littérature sont les réseaux proposés par Jordan [Jordan, 1990] et Elman [Elman, 1990], aussi appelés *Simple Recurrent Network* (SRN). Ils sont tous les deux caractérisés par des réseaux de neurones à 3 couches (une couche d'entrée, une couche cachée et une couche de sortie), de la forme simulateur à un pas. En revanche, ils sont différents dans leur manière de reboucler : alors que le réseau de Jordan reboucle les sorties de la couche de sortie en la considérant mesurable donc connue, les réseaux d'Elman rebouclent la sortie de la couche des neurones cachés. Ceci est illustré sur la figure 3.17 avec les boucles de récurrence représentées en orange. Ce type de réseau d'Elman a été développé pour l'analyse linguistique. Ces problèmes sont caractérisés par le fait que l'état du système n'a aucune raison d'être connu voire d'être déterminé par une loi physique donnée, comme des équations différentielles. De ce fait, il est inutile de le mesurer à la sortie du réseau puisque l'utilisateur est dans l'incapacité de la comparer à une mesure. Pour répondre à cela, le réseau d'Elman

représente donc l'état dans une couche cachée du réseau, c'est la raison pour laquelle il reboucle la sortie de la couche cachée et non de la couche de sortie en entrée du réseau.

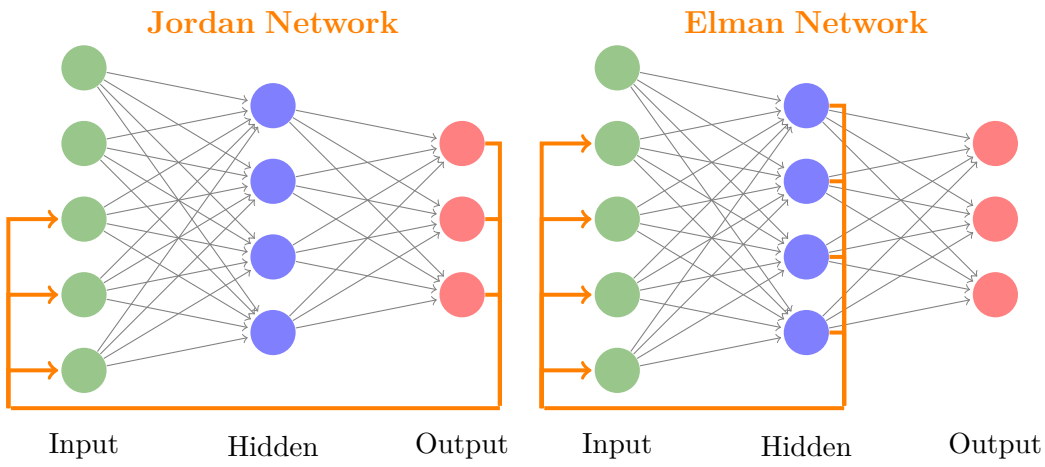


FIGURE 3.17 – Représentation des réseaux récurrents de Jordan et de Elman

Ensuite, des modèles du type perceptron multicouches présentés en section 3.2.2 rebouclés en simulateur (cf équation (3.32)) ont également été utilisés, ressemblant fortement à la structure d'un réseau de Jordan [Tutschku, 1995]. Leur application a notamment été réalisée dans le cadre de la thermique d'équipements électroniques par De Lozzo [Lozzo *et al.*, 2013].

Par ailleurs, il existe également des réseaux récurrents dont l'architecture est plus complexe que celle d'un perceptron multicouches rebouclé. Ils modifient notamment la structure et le fait d'aller d'un neurone d'une couche vers le neurone de la couche suivante. Cette idée, déjà introduite par Hopfield [Hopfield, 1982] dans lequel tous les neurones du réseau sont connectés les uns avec les autres (quel que soit leur couche), a ensuite été reprise dans le développement des *fully recurrent networks*.

Enfin, il existe également des réseaux de neurones à temps continu appelés *Continuous-Time Recurrent Neural Network* (CTRNN), dont le but était à l'origine de modéliser le fonctionnement d'un réseau de neurones biologique [Funahashi et Nakamura, 1993]. Mais ces réseaux sortent du cadre de cette étude qui se focalise sur la modélisation du phénomène de manière discrétisée en temps et en espace.

3.3.2.1 Apprentissage d'un réseau récurrent

Dans cette partie, il est question de l'apprentissage de perceptrons multicouches rebouclés comme à l'équation (3.32), c'est-à-dire utilisés comme un simulateur à un pas, comme dans l'étude de De Lozzo [Lozzo *et al.*, 2013]. Cette fois-ci, ce simulateur neuronal $\hat{f}$ est défini de la manière suivante :

$$\begin{array}{rcccl} \hat{f}\left(\bullet;\ \boldsymbol{w}\right) & : & \mathbb{R}^{N_u} \times \mathbb{R}^{N_y} & \longrightarrow & \mathbb{R}^{N_y} \\ & & \left(\boldsymbol{u}^{k-1}, \hat{\boldsymbol{y}}^{k-1}\right) & \longmapsto & \hat{\boldsymbol{y}}^{k} = \hat{f}\left(\boldsymbol{u}^{k-1}, \hat{\boldsymbol{y}}^{k-1};\ \boldsymbol{w}\right) \end{array}$$

Le fait que les réseaux de neurones rebouclés sont en fait composés d'un $\hat{f}$ indépendant du temps permet de se ramener à ce qui a été présenté à la partie 3.1. Toutefois, l'application

des techniques d'apprentissage statistique dans le cas rebouclé n'est pas directe et requiert la modification de :

— La définition du DOE, qui cette fois-ci n'est plus un ensemble de points, mais un ensemble de N_{DOE} trajectoires, avec N_t points de discrétisation temporelle pour chaque trajectoire :

$$DOE = \left\{ (\boldsymbol{u}^{k,i}, \boldsymbol{y}^{k,i})_{\substack{0 \leqslant k \leqslant N_t \\ 1 \leqslant i \leqslant N_{DOE}}} \right\} \tag{3.34}$$

— L'erreur d'apprentissage. Il est possible de voir les points à chaque pas de temps comme des observations supplémentaires par rapport au cas statique et donc de prendre l'erreur moindres carrés définie précédemment à l'équation (3.3) :

$$\mathcal{E}(\hat{f}\,(\bullet;\ \boldsymbol{w})\,, DOE) = \sum_{i=1}^{N_{DOE}} \sum_{k=1}^{N_t} \left\| \hat{f}(\boldsymbol{u}^{k-1,i}, \hat{\boldsymbol{y}}^{k-1,i};\ \boldsymbol{w}) - \boldsymbol{y}^{k,i} \right\|_2^2 \tag{3.35}$$

— Le gradient de l'erreur d'apprentissage. Sachant que le modèle $\hat{f}$ ne dépend pas du temps, une manière d'appliquer l'algorithme de rétropropagation du gradient est de « déplier » le réseau en temps pour obtenir un grand réseau équivalent mais uniquement composé des poids du réseau initial $\hat{f}$. Cette technique s'appelle la rétropropagation à travers le temps ou *Back Propagation Through Time* (BPTT) [Werbos, 1990]. Elle est d'ailleurs très utilisée du fait de sa rapidité d'exécution, notamment lorsqu'on la compare aux autres techniques existantes [Jaeger, 2002].

Une fois ces modifications réalisées, les techniques présentées dans la partie statique s'appliquent de la même manière que précédemment. Notamment, les algorithmes d'optimisation sont les mêmes ainsi que les techniques de sélection de modèle.

Autres techniques d'apprentissage de réseaux de neurones rebouclés. En dehors du BPTT qui est une technique reconnue pour sa faible complexité, il existe d'autres techniques qui peuvent être plus recommandées dans le cas de l'apprentissage en ligne (pour *on line* en anglais). Par exemple, le *Real-Time Recurrent Learning* consiste à mettre à jour le gradient de la sortie à chaque pas de temps de manière directe. Cette technique n'est toutefois pas adaptée dans le cas d'un apprentissage statique, en opposition au temps réel, puisque sa complexité est très grande par rapport à celle du BPTT [Jaeger, 2002].

Enfin, une technique provenant du milieu du filtrage est l'utilisation de filtre de Kalman étendu pour l'apprentissage de réseaux de neurones [Trebatický et Pospíchal, 2008], [Wang et Huang, 2011]. Le filtre de Kalman étendu est une technique d'estimation de systèmes non-linéaires [Julier et Uhlmann, 1997] dont le principe repose sur la linéarisation autour de l'état courant afin d'utiliser le filtre de Kalman linéaire classique en chaque état. Ces techniques permettent de donner une estimation de l'état d'un système dynamique à partir d'un état initial et d'observations à différents temps de discrétisation. Afin de l'adapter à l'apprentissage d'un réseau de neurones récurrents, il suffit de rechercher l'état optimal des poids $\boldsymbol{w}$ du réseau. Pour cela, ces poids sont vus comme l'état d'un système dynamique. Cet algorithme correspond à une technique de type gradient de descente d'ordre 2, au même titre que les optimisations par Levenberg-Marquardt présentées à la section 3.2.2.

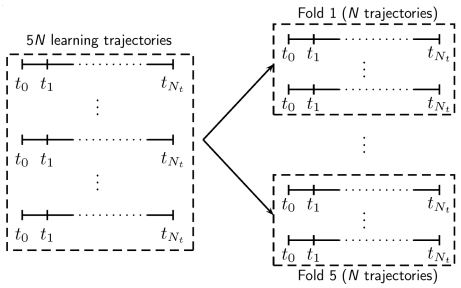


FIGURE 3.18 – Illustration de la répartition des trajectoires entières dans les *folds*

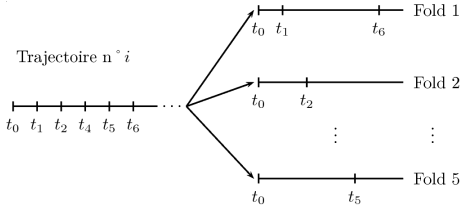


FIGURE 3.19 – Illustration de la construction des *folds* pour la validation croisée proposée par De Lozzo et consistant à répartir chaque pas de temps des trajectoires du DOE successivement dans les *folds*.

3.3.2.2 Sélection de modèle en spatio-temporel

Les techniques de sélection de modèle présentées dans le cas statique s'appliquent toujours en spatio-temporel, étant donné que le modèle $\hat{f}$ est supposé indépendant du temps. Cependant dans le cas de la validation croisée, il est nécessaire d'adapter la méthode afin de diviser le DOE en sous-espaces disjoints. Alors que dans le cas statique, ce sont des points à répartir, en spatio-temporel il s'agit de trajectoires. À la différence de points, qui ne peuvent être scindés, une trajectoire peut être considérée comme un tout, ou considérée comme un ensemble de points non liés entre eux. Dans le premier cas, illustré sur la figure 3.18, les *folds* sont conçus en répartissant les trajectoires entières. Dans le deuxième cas, chaque point de la trajectoire peut être vu comme une observation indépendante des autres points de la trajectoire pour le calcul des erreurs de test et d'apprentissage. Plus précisément, ceci signifie que deux pas de temps d'une même trajectoire peuvent être dans des *folds* différents. De cette manière, on peut les répartir dans les folds indifféremment. Ceci agrandit virtuellement la taille du DOE, par exemple dans le cas où il y a très peu de trajectoires, c'est-à-dire moins de cinquante. C'est ce qui est proposé dans l'étude de De Lozzo [Lozzo *et al.*, 2013]. Dans cette étude les points sont répartis dans les folds de manière successive, c'est-à-dire que le point $(\boldsymbol{u}^{1,i}, \boldsymbol{y}^{1,i})$ au pas de temps t^1 de chaque trajectoire $1 \leqslant i \leqslant N_{DOE}$ va dans le premier *fold*, le point $(\boldsymbol{u}^{2,i}, \boldsymbol{y}^{2,i})$ au pas de temps t^2 va dans le second *fold*, etc... L'avantage de cela est de fournir des *folds* équilibrés et possédant tous de l'information de chacune des trajectoires. Ceci est illustré en figure 3.19. En pratique pour construire chacun des V modèles pour la validation croisée, ceci implique qu'une même trajectoire sert à la fois pour minimiser l'erreur d'apprentissage et pour calculer l'erreur de validation croisée. Ce ne sont toutefois pas les mêmes zones de la trajectoire qui sont utilisés pour réaliser chacune de ces actions.

3.3.3 Modélisations basées sur les processus gaussiens

En dehors des techniques basées sur des modèles récurrents provenant de l'identification de système, il existe des généralisations du krigeage au cas où les entrées et sorties dépendent du temps. Une généralisation récente proposée par Morris [Morris, 2012] repose sur la modification du noyau de corrélation R au cas dynamique. Pour rappel, dans le cas statique, ce noyau pouvait être de la forme suivante :

$$R(\boldsymbol{x}, \boldsymbol{x}';\ \boldsymbol{w}) = \exp\left(-d(\boldsymbol{x}, \boldsymbol{x}';\ \boldsymbol{w})\right)$$

Par exemple dans le cas gaussien à l'équation (3.18), $d(\boldsymbol{x}, \boldsymbol{x}';\ \boldsymbol{w}) = \sum_{i=1}^{N_x} \left(\frac{x_i - x_i'}{w_i}\right)^2$. Afin de prendre en compte le temps, Morris modifie le calcul de cette distance dans le cas où $\boldsymbol{x}$ est une fonction du temps. Dans notre cas, cette fonction du temps est l'entrée exogène $\boldsymbol{u}(t)$. Dans le cas d'une seule entrée exogène, cette modification s'écrit de la manière suivante :

$$\mathcal{D}(u, u', T;\ \boldsymbol{w}) = \int_0^T \tau(T - t;\ \boldsymbol{w}) d(u(t), u'(t))\ dt \tag{3.36}$$

T étant le temps où l'on souhaite connaître la valeur de la sortie. En paramétrant la fonction $\tau\left(\bullet;\ \boldsymbol{w}\right) : \mathbb{R} \longrightarrow \mathbb{R}$ par nos paramètres $\boldsymbol{w}$, dont la définition reste donc inchangée. La forme de cette fonction τ permet de prendre en compte toute sorte d'*a priori* sur le phénomène à prédire, comme une sensibilité moins importante aux entrées avec le temps. Ceci est par exemple le cas du passage d'un état transitoire vers un état permanent. Une forme adaptée à cela serait :

$$\tau(T - t;\ \boldsymbol{w}) = \exp\left(-w_1 (T - t)^2\right) \text{ avec } w_1 > 0$$

Si l'*a priori* est faible, il est conseillé d'utiliser une forme paramétrique pour τ très générale. Par ailleurs, ce noyau de corrélation peut de la même manière être généralisé au cas spatio-temporel : soit $p \in [p_L, p_U]$ la position spatiale et (q, T) la position et le temps où l'on souhaite prédire la sortie :

$$\mathcal{D}(u, u', q, T;\ \boldsymbol{w}) = \int_0^T \int_{p_L}^{p_U} \tau(q - p, T - t;\ \boldsymbol{w}) d(u(p, t), u'(p, t))\ dtdp$$

En ce qui concerne l'apprentissage, le maximum de vraisemblance peut facilement être généralisé à cette dépendance en temps. Comme pour les réseaux de neurones rebouclés, il se fait à partir de couples entrées exogènes/sorties discrétisés en temps : le maximum de vraisemblance total est donc la somme des maxima de vraisemblance en chaque pas de temps.

Cette construction repose cependant sur certaines hypothèses :

1. Elle traite les sorties à prédire entre deux pas de temps différents de manière indépendante. En effet, la corrélation entre deux profils u et u' se ramène à :

$$\begin{aligned} \text{corr}\left(u, u', q, T\right) &= \exp\left\{-w_0 \int_0^T \int_{p_L}^{p_U} \tau(q - p, T - t;\ \boldsymbol{w}) d(u(p, t), u'(p, t))\ dtdp\right\} \\ &= \exp\left\{-w_0 \mathcal{D}(u, u', q, T;\ \boldsymbol{w})\right\} \end{aligned} \tag{3.37}$$

   On remarque donc que cette corrélation se calcule en chaque point (q, T), elle est donc nulle entre deux temps $T \neq T'$. En particulier, ceci signifie que la prédiction

du comportement en temps de la sortie correspond davantage à une interpolation entre les profils en entrée en chaque couple (q, T) plutôt qu'à une interpolation au cours du temps.

2. Le choix de la fonction de noyau τ est crucial et peut nécessiter un *a priori* important.
3. Le nombre de points discrétisés dans le DOE peut être très important, or ce dernier donne directement la taille de la matrice $\mathbf{\Gamma}$ à inverser afin d'estimer le modèle $\hat{f}$. Ceci implique donc que le coût d'estimation du modèle de substitution peut être important, surtout si elle est comparée à la parcimonie des réseaux de neurones récurrents.

Chapitre 4

Quantification des effets des incertitudes en entrée d'un modèle boîte noire

Sommaire

Le problème industriel présentant de nombreuses incertitudes, l'objectif de cette partie est de dresser un état de l'art des techniques permettant la propagation des incertitudes à travers un modèle déterministe $\phi : \mathbb{R}^{N_\xi} \longrightarrow \mathbb{R}$.

Soit $\boldsymbol{\xi} = \left(\xi_1, \cdots, \xi_{N_\xi}\right)$ le vecteur de variables aléatoire en entrée de ce modèle et soit $\mathcal{D}_\xi \subset \mathbb{R}^{N_\xi}$ le support de ces variables aléatoires. Chacune des composantes ξ_i est distribuée suivant une loi de probabilité (définies à l'annexe A) de densité h_i et l'on fait l'hypothèse que ses composantes sont indépendantes deux à deux, donc la densité jointe des entrées est le produit des densités marginales $h = \Pi_{i=1}^{N_\xi} h_i$. Même si le modèle ϕ est déterministe, si ces entrées sont des variables aléatoires alors sa sortie est également une variable aléatoire $Y = \phi(\boldsymbol{\xi})$, comme illustré sur la figure 4.1.

Par ailleurs, les ingénieurs fixent souvent de manière déterministe des paramètres dont ils ne connaissent pas précisément la valeur, soit parce que l'aléa est intrinsèque au phénomène que traduit ce paramètre, soit par manque de connaissances. Les incertitudes du premier type sont les incertitudes aléatoires et sont irréductibles, celles du second type sont les incertitudes épistémiques qui peuvent être réduites en faisant davantage d'efforts

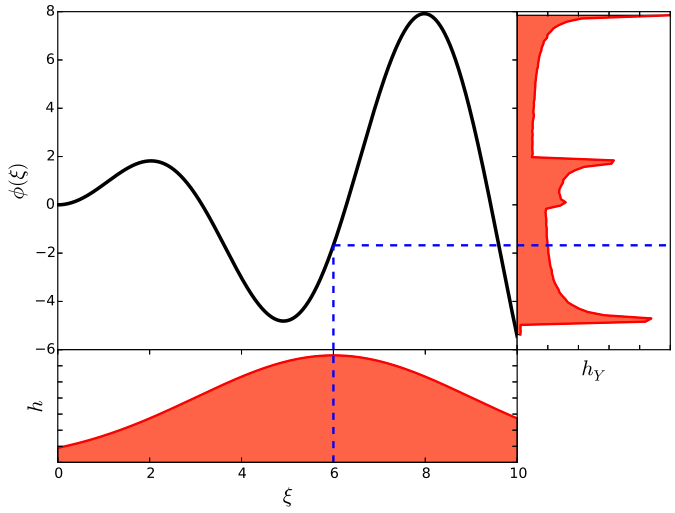


FIGURE 4.1 – Densité de la sortie d'un modèle déterministe ϕ avec une entrée suivant une loi normale tronquée

(de calcul, de retour d'expériences ou encore de mesure). Lorsqu'un paramètre incertain n'est pas traité comme tel, cela revient par exemple à utiliser l'espérance de ce paramètre en entrée afin de pouvoir obtenir la valeur de la sortie du modèle. Or, la figure 4.1 montre que ceci peut engendrer des erreurs importantes. En effet, le trait en pointillé illustre la moyenne de la distribution d'entrée puis sa propagation à travers le modèle déterministe qui fournit sa position par rapport à la distribution de la sortie. Cela permet de montrer que la sortie de l'espérance de l'entrée ne correspond ni à l'espérance de la sortie, ni au maximum de la densité.

Pour prendre en compte l'effet des incertitudes sur la sortie, il est donc nécessaire de travailler avec une variable aléatoire. Néanmoins, c'est une quantité qui peut être difficile à appréhender et qui de plus n'apporte pas une information facilement manipulable, aussi bien par l'ingénieur que par des algorithmes d'optimisation. De ce fait, il est nécessaire d'introduire des mesures de robustesse liés au caractère probabiliste de l'entrée et permettant de quantifier un effet précis sur la distribution de la sortie $Y = \phi(\xi)$. La première partie a donc pour but d'introduire ces mesures de robustesse. Ensuite, il existe différentes manières d'estimer ces paramètres. La deuxième partie consiste donc à en expliciter certaines puis à expliquer comment en réduire le nombre d'appels à la fonction déterministe par l'usage de modèles de substitution au cours de l'estimation.

D'autre part, si la fonction est pratiquement constante par rapport à un paramètre, ceci signifie que sa distribution de sortie en dépend peu. Ceci illustre le fait qu'il n'y pas toujours un intérêt à propager les incertitudes autour d'un paramètre. Plus précisément, il faut vérifier que les variables entachées d'incertitudes influent sur la sortie. Pour cela, les méthodes d'analyse de sensibilité sont présentées dans la dernière section 4.3. Elles permettent de déterminer quels paramètres sont influents, et lesquels ne le sont pas, en quantifiant l'impact des incertitudes en entrée sur la variabilité de la sortie.

4.1 Introduction de mesures de robustesse d'une variable aléatoire

Une variable aléatoire est une quantité difficilement manipulable, il est donc nécessaire d'introduire des mesures permettant d'en extraire des informations. Elles sont appelées mesures de robustesse, notées ici $\rho_\xi(\phi)$ et, selon l'effet des incertitudes que souhaite quantifier l'ingénieur, il en existe de différentes sortes.

Par exemple, les mesures les plus intuitives et qui sont le plus fréquemment utilisées sont l'espérance et la variance, définies de la manière suivante :

$$\rho_\xi(\phi) = \mathbb{E}\left[\phi(\boldsymbol{\xi})\right] = \int_{\boldsymbol{\chi}\in\mathcal{D}_\xi} \phi(\boldsymbol{\chi}) h(\boldsymbol{\chi}) d\boldsymbol{\chi} \tag{4.1}$$

$$\rho_\xi(\phi) = \mathrm{Var}\left[\phi(\boldsymbol{\xi})\right] = \mathbb{E}\left[\Big(\phi(\boldsymbol{\xi}) - \mathbb{E}\left[\phi(\boldsymbol{\xi})\right]\Big)^2\right] = \int_{\boldsymbol{\chi}\in\mathcal{D}_\xi} \left(\phi(\boldsymbol{\chi}) - \mathbb{E}\left[\phi(\boldsymbol{\xi})\right]\right)^2 h(\boldsymbol{\chi}) d\boldsymbol{\chi} \tag{4.2}$$

L'espérance représente la valeur moyenne de la sortie, la variance représente l'écart quadratique moyen à cette moyenne. En d'autres termes, l'espérance peut être vue comme une sortie « débruitée » alors que la variance quantifie justement la valeur moyenne du bruit que l'on filtre. Toutefois, ce sont des mesures en moyenne et, dans un cadre général, elles ne donnent aucune information sur les phénomènes non moyens, comme des queues de distribution. Ceci est illustré sur la figure 4.2 : dans le cas d'une distribution du type bimodale, ou à queue lourde, ces mesures ne sont pas représentatives de la distribution. Notamment, elles ne fournissent pas toute l'information sur la queue de distribution. Or, ce sont justement ces niveaux qui permettent de mesurer le risque et donc de traduire une aversion à ce risque. Sachant que, dans le cadre de cette étude, ce sont justement des mesures de risque qui nous intéressent puisque l'on ne veut pas sous-estimer l'effet des incertitudes sur la sortie, il est nécessaire d'introduire d'autres mesures de robustesse. Rockafellar [Rockafellar, 2007] a fait une revue des mesures de risque dont voici la liste :

- le pire cas $\rho_\xi(\phi) = \max_\xi(\phi(\xi))$. C'est une mesure déterministe dans le sens où les incertitudes sont uniquement traitées par l'intervalle dans lequel elles peuvent prendre leur valeur, par exemple $\xi \in [a, b]$. En revanche, du fait que les lois des incertitudes en entrée ne sont pas prises en compte, c'est-à-dire la probabilité d'occurrence de chacune des valeurs possibles, le voisinage du point ξ où tombe ce maximum peut avoir une densité de probabilité d'occurrence très faible. Cette mesure peut donc être trop conservative. D'où l'intérêt des mesures probabilistes qui permettent de moins considérer les zones de faibles probabilités. La loi de probabilité des entrées est inutile dans le but d'estimer un pire cas, le calcul de cette mesure de robustesse se ramène donc à un problème d'optimisation déterministe classique dont la résolution est développée en section 5.1.
- α-quantile $\rho_\xi(\phi) = q_\alpha(\phi)$ avec $q_\alpha(\phi)$ tel que $Pr_\xi(\phi(\xi) \leqslant q_\alpha(\phi)) = \alpha$. Cette mesure a pour limite de ne donner aucune information sur les $1-\alpha\%$ des cas les plus extrêmes puisqu'elle ne les prend pas en compte.
- α-superquantile $\rho_\xi(\phi) = Q_\alpha$. C'est une mesure, provenant de la littérature appliquée à la finance, qui permet de quantifier le poids des évènements au dessus de l'α-quantile. Il est défini par Rockafellar comme :

$$Q_\alpha(\phi) = \frac{1}{1-\alpha} \int_\alpha^1 q_u(\phi)\ du \tag{4.3}$$

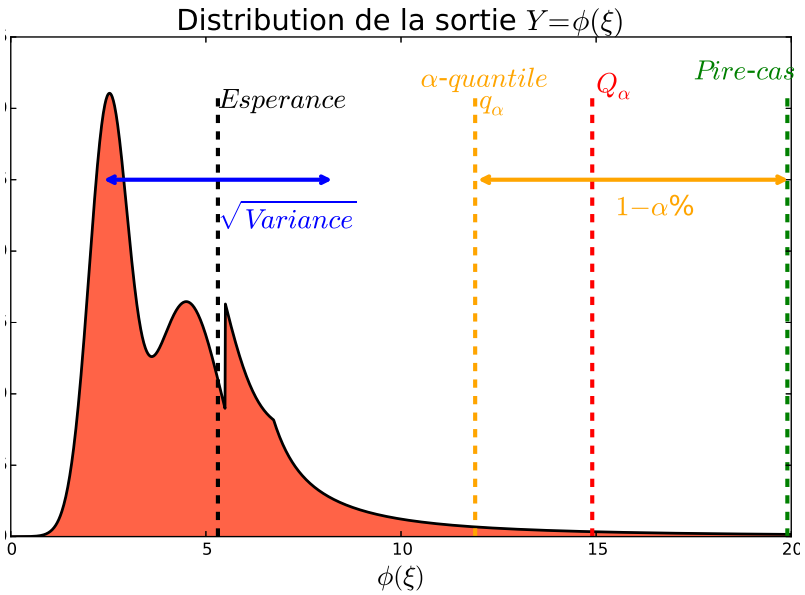


FIGURE 4.2 – Illustration des mesures de robustesse estimant les queues de distribution

Cette mesure est une moyenne des u-quantiles avec $u \geqslant \alpha$. C'est donc une extension du quantile car elle pondère tous les évènements situés entre ce dernier et le pire cas.

Au niveau des propriétés mathématiques, le superquantile est plus intéressant que le quantile. En effet, Rockafellar [Rockafellar, 2007] a montré que le superquantile avait l'avantage d'être une mesure de risque cohérente, ce que le quantile n'est pas. Ceci signifie qu'il est caractérisé par les propriétés mathématiques suivantes :

1. Monotonie : soit ϕ_1 et ϕ_2 deux fonctions déterministes telles que $\forall \boldsymbol{\chi} \in \mathcal{D}_\xi$: $\phi_1(\boldsymbol{\chi}) \leqslant \phi_2(\boldsymbol{\chi})$, alors $\rho_\xi(\phi_1) \leqslant \rho_\xi(\phi_2)$.
2. Convexité : soit ϕ_1 et ϕ_2 deux fonctions déterministes et soit $t \in [0,1]$ alors :
$$\rho_\xi(t\phi_1 + (1-t)\phi_2) \leqslant t\rho_\xi(\phi_1) + (1-t)\rho_\xi(\phi_2)$$
3. Équivariant en translation : soit $a \in \mathbb{R}$, alors $\rho_\xi(\phi_1 + a) = \rho_\xi(\phi_1) + a$.
4. Homogénéité positive : soit $a > 0$, alors $\rho_\xi(a\phi_1) = a\rho_\xi(\phi_1)$.

En dehors du cas de distributions particulières, le quantile ne satisfait pas en général la condition 2 de convexité. Or cette propriété est avantageuse dans le cadre d'un problème d'optimisation où toute propriété de régularité mathématique supplémentaire permet d'envisager davantage d'algorithmes de résolution.

Remarque sur l'utilisation de mesures mélangeant moyenne et variance pour la mesure de risque : Sachant que l'espérance et la variance sont deux des mesures de robustesse les moins coûteuses à estimer, certaines études proposent de borner voire d'estimer les mesures de risque présentées ici à l'aide de ces quantités. Par exemple, Padulo et Guenov [Padulo et Guenov, 2011] approchent le quantile et le superquantile (qu'ils appellent *Tail Conditional Expectation*) à l'aide de formules du type :

$$\rho_\xi = \mathbb{E}\left[\phi(\boldsymbol{\xi})\right] + k(h_Y)\ \sqrt{\mathrm{Var}\left[\phi(\boldsymbol{\xi})\right]}$$

avec k dépendant de la loi de la sortie h_Y. Cette approximation est valable dans le cas où la distribution de la sortie appartient à certaines familles de lois connues analytiquement. Ceci présente l'avantage de réduire le coût d'estimation de ces mesures de risque de manière importante. Néanmoins, l'hypothèse sur la distribution de sortie est forte et n'est pas vérifiée dans le cas général. Par exemple, la figure 4.1 montre que même avec une loi normale en entrée d'une fonction 1D régulière, la distribution de sortie n'appartient à aucune loi connue. La borne proposée par l'étude de Padulo et Guenov est dans ces cas-là inutilisable.

4.2 Estimation des mesures de risque

Dans cette partie, il est question de l'estimation des mesures de type quantile et superquantile à l'aide de méthodes numériques. De ce fait, beaucoup de techniques qui ne sont adaptées qu'à l'estimation de mesures exprimables sous la forme d'une espérance (comme la variance ou les probabilités de dépassement de seuil) ne sont ici pas utilisables. Par exemple, la méthode des moments [Padulo et Guenov, 2011] ou les polynômes de chaos [Blatman et Sudret, 2011] [Rajabi *et al.*, 2015] sans simulations Monte-Carlo ne sont pas adaptées à l'estimation de ces mesures de risque. En effet, ces modèles de substitution ne permettent d'évaluer analytiquement que l'espérance et la variance de la sortie.

Les méthodes de Monte-Carlo étant les plus répandues, leur utilisation dans le but d'estimer un quantile ou un superquantile sont présentées. Toutefois, ces mesures ont pour vocation d'être utilisées dans le cadre d'un problème d'optimisation. Il est donc primordial que le nombre d'évaluations nécessaires à la fonction déterministe soit réduit et la méthode de Monte-Carlo nécessite trop d'appels afin d'obtenir une précision suffisante lorsqu'elle est utilisée sur des quantiles. Pour diminuer la taille de l'échantillon requise pour une bonne précision d'estimation, les méthodes permettant de réduire la variance sont présentées. L'échantillonnage préférentiel (appelé *Importance sampling* en anglais) ou l'échantillonnage stratifié (appelé *Importance splitting* en anglais) en sont deux exemples.

Enfin, afin de réduire davantage le nombre d'appels nécessaires à la fonction ϕ afin d'obtenir des estimateurs de précision suffisante, l'assistance par krigeage de ces méthodes de Monte-Carlo modifiées est détaillée.

4.2.1 Limite des estimateurs par méthodes de Monte-Carlo

La méthode de Monte-Carlo consiste à tirer un échantillon de N_{MC} points $\left(\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^{N_{MC}}\right)$ à partir de la loi h des entrées afin d'estimer les diverses mesures de robustesse.

Estimation du quantile : sachant que sa définition mathématique est basée sur la connaissance de la fonction de répartition H, l'estimateur par Monte-Carlo du quantile est donc basé sur l'approximation de la fonction de répartition $\tilde{H}_{N_{MC}}$ par Monte-Carlo :

$$\left\{ \begin{array}{rcl} \tilde{H}_{N_{MC}}(\boldsymbol{\chi}) & = & \dfrac{1}{N_{MC}} \displaystyle\sum_{i=1}^{N_{MC}} \mathbf{1}_{\phi(\boldsymbol{\xi}^i) \leqslant \boldsymbol{\chi}} \\ \tilde{q}_\alpha^{MC} & = & \min\left\{ \boldsymbol{\chi} \ : \ \tilde{H}_{N_{MC}}(\boldsymbol{\chi}) \geqslant \alpha \right\} \end{array} \right. \tag{4.4}$$

D'après [David et Nagaraja, 2004], on peut montrer que l'erreur commise par cet estimateur se comporte asymptotiquement de la manière suivante :

$$\sqrt{N_{MC}} \left(\tilde{q}_\alpha^{MC} - q_\alpha \right) \underset{N_{MC}\infty}{\xrightarrow{\mathcal{L}}} \sigma_{q,MC} \mathcal{N}(0,1) \text{ avec } \sigma_{q,MC} = \frac{\sqrt{\alpha(1-\alpha)}}{h_Y(q_\alpha)} \tag{4.5}$$

avec h_Y la densité de la sortie. Ce résultat est démontré dans la partie suivante dans le cas plus général de l'échantillonnage préférentiel. Sa démonstration repose sur l'utilisation du théorème de Berry-Esseen. Or, la densité a dans le cas général une valeur faible au niveau du quantile, ce qui se traduit par un $h_Y(q_\alpha) \underset{\alpha \to 1}{\to} 0$. Ceci implique que la taille de

l'échantillon Monte-Carlo N_{MC} doit être grande afin que sa racine carrée compense ce comportement. Par exemple, il est rarement envisageable d'utiliser moins de 10^4 échantillons pour un quantile de l'ordre de $\alpha \simeq 10^{-2}$.

Estimation du superquantile : Une manière alternative de calculer le superquantile, équivalente à sa définition, est proposée par Rockafellar [Rockafellar et Uryasev, 2000]. Elle est donnée par la formule variationnelle suivante :

$$Q_\alpha = \inf_{\gamma\in\mathbb{R}} \Psi(\gamma) = \inf_{\gamma\in\mathbb{R}} \left[\gamma + \frac{1}{1-\alpha} \int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}} \mathbf{1}_{\phi(\boldsymbol{\chi})\geqslant\gamma} \left(\phi(\boldsymbol{\chi}) - \gamma\right) h(\boldsymbol{\chi}) d\boldsymbol{\chi} \right] \tag{4.6}$$

L'une des valeurs de γ où se réalise le minimum de la fonction $\Psi(\gamma)$ est égale à la valeur de q_α. Plus précisément il s'agit de la plus petite valeur de γ où se réalise ce minimum. Dans le cas où l'on suppose que la distribution de la sortie est continue, ceci provient du fait que $\Psi(\gamma)$ est convexe et continûment différentiable et que l'équation d'optimalité (annulation du gradient) est satisfaite par le quantile. Enfin, en posant $\gamma = q_\alpha$ on retrouve la formule du superquantile de l'équation (4.3). Dans le cas plus général où la distribution de sortie est continue par morceaux (contenant donc des sauts), Rockafellar et Uryasev ont fourni une preuve de convergence dans l'étude [Rockafellar et Uryasev, 2002].

En évaluant l'intégrale par la méthode de Monte-Carlo, on obtient donc comme estimateur Monte-Carlo :

$$\tilde{Q}_\alpha^{MC} = \inf_{\gamma\in\mathbb{R}} \left[\gamma + \frac{1}{N_{MC}\,(1-\alpha)} \sum_{i=1}^{N_{MC}} \mathbf{1}_{\phi(\boldsymbol{\xi}^i)\geqslant\gamma} \left(\phi(\boldsymbol{\xi}^i) - \gamma\right) \right] \tag{4.7}$$

Au niveau de l'erreur d'estimation, on a, d'après [Hong *et al.*, 2014] :

$$\sqrt{N_{MC}} \left(\tilde{Q}_\alpha^{MC} - Q_\alpha\right) \xrightarrow[N_{MC}\infty]{\mathcal{L}} \sigma_{Q,MC}\mathcal{N}(0,1) \text{ avec } \sigma_{Q,MC} = \frac{\sqrt{\mathrm{Var}\left(\mathbf{1}_{\phi(\xi)\geqslant q_\alpha}\left(\phi(\xi) - q_\alpha\right)\right)}}{(1-\alpha)} \tag{4.8}$$

Comme précédemment pour le quantile, cette variance nécessite un nombre d'échantillons N_{MC} grand afin d'être petite. En effet, la valeur de $1-\alpha$ au dénominateur a tendance à être petite, de l'ordre de 10^{-2}, il est donc nécessaire de compenser suffisamment cela par la racine carrée du nombre de points. Une fois encore, il est inenvisageable d'utiliser un nombre de points $N_{MC} < 10^4$

4.2.2 Estimateurs par méthodes de Monte-Carlo à variance réduite

Dans le but de réduire la taille de l'échantillon nécessaire à l'estimation du quantile et du superquantile avec un précision suffisante, il existe deux principales stratégies d'échantillonnage notamment développées dans [Morio et Balesdent, 2015] et [Caron *et al.*, 2014] : la première est l'échantillonnage stratifié et repose sur la décomposition de l'estimation du quantile en l'estimation de probabilités conditionnelles estimées sur des sous-ensembles définis itérativement. La seconde est l'échantillonnage préférentiel et repose sur le tirage des échantillons à l'aide d'une loi biaisée afin que ces échantillons soient générés dans la zone d'intérêt, et ainsi diminuer la variance de l'estimateur. La difficulté dans le cas de l'échantillonnage préférentiel réside dans le choix de la loi biaisée et un algorithme séquentiel est détaillé.

4.2.2.1 Échantillonnage stratifié

Initialement développée afin d'estimer des probabilités de dépassement de seuil $\mathbb{P}\left(\phi(\boldsymbol{\xi}) > S\right)$, cette méthode d'échantillonnage s'applique également à l'estimation de quantile [Guyader *et al.*, 2011] [Morio et Balesdent, 2015]. Son principe repose sur la décomposition du problème en sous-problèmes plus faciles à résoudre. Pour ce faire, l'idée est de décomposer la probabilité à estimer en probabilités conditionnelles à l'aide de la formule de Bayes. Soit Ω l'évènement dont on souhaite calculer la probabilité, par exemple ici $\Omega = \{\boldsymbol{\xi} : \phi\left(\boldsymbol{\xi}\right) > S\}$. Afin de décomposer le calcul de cette probabilité il faut introduire une séquence décroissante d'évènements de la manière suivante : $\Omega_1 \supset \Omega_2 \supset \cdots \supset \Omega_m = \Omega$. Cette séquence peut en particulier être définie de la manière suivante : $\Omega_i = \{\boldsymbol{\xi} : \phi\left(\boldsymbol{\xi}\right) > S_i\}$ avec $S_1 < S_2 < \cdots < S_m = S$. Alors $\mathbb{P}(\boldsymbol{\xi} \in \Omega)$ se décompose ainsi :

$$\begin{aligned}
\mathbb{P}(\boldsymbol{\xi} \in \Omega) &= \mathbb{P}\left(\boldsymbol{\xi} \in \cap_{i=1}^{m} \Omega_i\right) \\
&\underset{\text{Bayes}}{=} \mathbb{P}\left(\boldsymbol{\xi} \in \Omega_m \;\middle|\; \boldsymbol{\xi} \in \cap_{i=1}^{m-1} \Omega_i\right) \mathbb{P}\left(\boldsymbol{\xi} \in \cap_{i=1}^{m-1} \Omega_i\right) \\
&\underset{\text{def } \Omega_i}{=} \mathbb{P}\left(\boldsymbol{\xi} \in \Omega_m \mid \boldsymbol{\xi} \in \Omega_{m-1}\right) \mathbb{P}\left(\boldsymbol{\xi} \in \cap_{i=1}^{m-1} \Omega_i\right) \\
&= \cdots \\
&\underset{\text{récurrence}}{=} \mathbb{P}\left(\boldsymbol{\xi} \in \Omega_1\right) \Pi_{i=1}^{m-1} \mathbb{P}\left(\boldsymbol{\xi} \in \Omega_{i+1} \mid \boldsymbol{\xi} \in \Omega_i\right)
\end{aligned} \tag{4.9}$$

Afin d'estimer les probabilités conditionnelles $\mathbb{P}\left(\boldsymbol{\xi} \in \Omega_{i+1} \mid \boldsymbol{\xi} \in \Omega_i\right)$ par Monte-Carlo, il est nécessaire de générer des points selon la densité conditionnelle h_i :

$$h_i = \frac{\mathbf{1}_{\boldsymbol{\xi} \in \Omega_i} h(\boldsymbol{\xi})}{\mathbb{P}\left(\boldsymbol{\xi} \in \Omega_i\right)} = \frac{\mathbf{1}_{\phi(\boldsymbol{\xi}) \geqslant S_i} h(\boldsymbol{\xi})}{\mathbb{P}\left(\phi\left(\boldsymbol{\xi}\right) \geqslant S_i\right)} \tag{4.10}$$

Cette génération peut se faire à l'aide de l'algorithme de Metropolis-Hastings [Chib et Greenberg, 1995]. C'est un algorithme basé sur la théorie des chaînes de Markov, donc itératif, permettant de générer une densité inconnue. Elle consiste principalement à faire des tirages par acceptation-rejet à partir de l'état courant.

En ce qui concerne les seuils S_i, le choix diminuant le plus la variance est celui dont les probabilités conditionnelles obtenues sont constantes [Pastel *et al.*, 2014]. De ce fait, le β-quantile empirique des échantillons générés selon la loi conditionnelle est utilisé avec β à fixer par l'utilisateur. Ceci permet d'obtenir :

$$\mathbb{P}\left(\boldsymbol{\xi} \in \Omega_{i+1} \mid \boldsymbol{\xi} \in \Omega_i\right) = 1 - \beta$$

Le choix de ce β est par ailleurs très sensible puisqu'un β trop faible implique une convergence lente alors qu'un β trop grand induit un biais d'estimation important.

L'inconvénient de cette méthode est le nombre important de paramètres à fixer. Par exemple, le nombre de points n_{ISp} par estimation de $\mathbb{P}\left(\boldsymbol{\xi} \in \Omega_{i+1} \mid \boldsymbol{\xi} \in \Omega_i\right)$, la valeur de β, ou encore les paramètres que nécessitent l'algorithme de Metropolis-Hastings.

Estimation du quantile par stratification : Dans le cas de l'estimation d'un α-quantile et non plus d'une probabilité de dépassement de seuil, la probabilité $\mathbb{P}(\boldsymbol{\xi} \in \Omega) = 1-\alpha$ est connue mais c'est la valeur du seuil S qui est recherchée. Grâce à cela, il est possible de fixer le seuil de quantile empirique β dès que le nombre de strates m souhaité par l'utilisateur est connu. En effet, $\alpha = 1 - \mathbb{P}(\boldsymbol{\xi} \in \Omega) = 1 - (1-\beta)^m$. Or α et m sont connus,

Algorithme 4.1 Échantillonnage stratifié pour l'estimation de quantile

Input : ϕ la fonction à évaluer
h la densité jointe des incertitudes en entrée
α la probabilité du quantile à calculer
Le nombre de strates m
n_{Isp} le nombre de point par strates

Output : le quantile $\tilde{q}_\alpha^{ISp}$

· Calculer $\beta = 1 - \exp(\frac{\ln(1-\alpha)}{m})$;
· Générer n_{ISp} échantillons selon h $\left(\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^{n_{ISp}}\right)$;
· Calculer la valeur des sorties en ces points $\left(\phi(\boldsymbol{\xi}^1), \cdots, \phi(\boldsymbol{\xi}^{n_{ISp}})\right)$;
· Évaluer S_1 le quantile empirique de cet échantillon;
for $k \in \{2, \cdots, m\}$ **do**
  · Générer les échantillons selon la loi conditionnelle h_{k-1} : $\left(\boldsymbol{\xi}^{1,k}, \cdots, \boldsymbol{\xi}^{n_{ISp},k}\right)$
  · Calculer la valeur des sorties en ces points ϕ $\left(\phi(\boldsymbol{\xi}^{1,k}), \cdots, \phi(\boldsymbol{\xi}^{n_{ISp},k})\right)$;
  · Évaluer S_k le quantile empirique de cet échantillon;
end
$\tilde{q}_\alpha^{ISp} = S_m$

on a donc pour $\beta = 1 - \exp(\frac{\ln(1-\alpha)}{m})$. Par exemple pour un quantile à $\alpha = 99\%$ et $m = 5$ strates, $\beta \simeq 0.602$. Ceci est détaillé par l'algorithme 4.1. En ce qui concerne son erreur d'estimation, en prenant $\beta = \frac{1}{n_{Isp}}$, [Caron *et al.*, 2014] appelle cet algorithme *Last Particle* et a montré que son estimateur avait une erreur qui a le comportement asymptotique suivant :

$$\sqrt{N}\left(\tilde{q}_\alpha^{Isp} - q_\alpha\right) \xrightarrow[n_{Isp}\infty]{\mathcal{L}} \sigma_{Q,Isp}\,\mathcal{N}(0,1) \text{ avec } \sigma_{Q,Isp} = \frac{-(1-\alpha)^2\ln(1-\alpha)}{h_Y(q_\alpha)} \tag{4.11}$$

La figure 4.3 montre que dans le cas où l'on estime un quantile α suffisamment grand, alors la diminution de variance est de plus en plus effective.

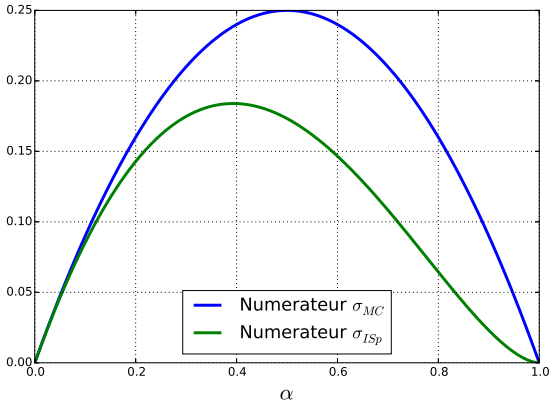


FIGURE 4.3 – Comparaison du numérateur de la variance de l'erreur commise dans l'estimation du quantile entre la méthode de Monte-Carlo et la méthode par stratification

Estimation du superquantile par stratification : il n'existe à notre connaissance pas de travail utilisant la stratification pour l'estimation de superquantile.

4.2.2.2 Échantillonnage préférentiel

L'idée de l'échantillonnage préférentiel est de tirer les variables aléatoires d'entrée selon une densité biaisée h_b à la place de la vraie loi h, de sorte à favoriser les points qui correspondent à l'évènement que l'on recherche. Ceci est illustré sur la figure 4.4 : dans le cas de l'estimation d'un quantile ou d'un superquantile, une densité biaisée favorisant la queue de distribution permettrait de diminuer le nombre d'appels à la fonction ϕ pour une précision équivalente sur le résultat.

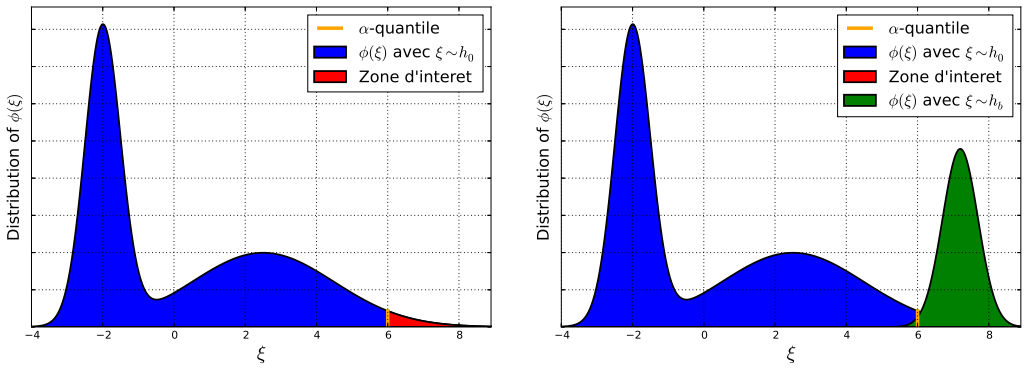


FIGURE 4.4 – Illustration de la loi de la sortie selon l'entrée et d'un exemple de loi biaisée h_b satisfaisante

Puisque les échantillons sont tirés selon une loi biaisée, il est indispensable de corriger les estimateurs d'intégrales par Monte-Carlo afin de ne pas introduire de biais. Ce terme correcteur s'appelle rapport de vraisemblance et vaut pour un échantillon $\tilde{\boldsymbol{\xi}}$ tiré par la loi biaisée h_b :

$$L(\tilde{\boldsymbol{\xi}}) = \frac{h(\tilde{\boldsymbol{\xi}})}{h_b(\tilde{\boldsymbol{\xi}})} \tag{4.12}$$

Il provient en fait d'un simple changement de mesure probabiliste dans l'estimation des intégrales, par exemple pour le calcul de l'espérance de $Y = \phi(\xi)$:

$$\begin{aligned} \mathbb{E}_h\left[Y = \phi(\xi)\right] &= \int_{\mathcal{D}_\xi} \phi(\boldsymbol{\chi}) h(\boldsymbol{\chi}) d\boldsymbol{\chi} \\ &= \int_{\mathcal{D}_\xi} \phi(\boldsymbol{\chi}) \frac{h(\boldsymbol{\chi})}{h_b(\boldsymbol{\chi})}\, h_b(\boldsymbol{\chi}) d\boldsymbol{\chi} \\ &= \mathbb{E}_{h_b}\left[\phi(\xi)\frac{h(\boldsymbol{\xi})}{h_b(\boldsymbol{\xi})}\right] = \mathbb{E}_{h_b}\left[\phi(\xi) L(\boldsymbol{\xi})\right] \end{aligned} \tag{4.13}$$

Estimateur du quantile par échantillonnage préférentiel : Dans le cas du quantile, on peut donc reprendre les formules rappelées dans le cas de l'estimateur Monte-Carlo et les corriger par le rapport de vraisemblance pour chaque échantillon. Soit $\left(\tilde{\boldsymbol{\xi}}^1, \cdots, \tilde{\boldsymbol{\xi}}^{N_{IS}}\right)$ les N_{IS} points générés selon la loi biaisée h_b. Voici l'estimateur par échantillonnage préférentiel

du quantile :

$$\left\{ \begin{array}{rcl} \tilde{H}_{N_{IS}}(\boldsymbol{\chi}) & = & \dfrac{1}{N_{IS}} \displaystyle\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \leqslant \boldsymbol{\chi}} \\ \tilde{q}_\alpha^{IS} & = & \min \left\{ \boldsymbol{\chi} \ : \ \tilde{H}_{N_{IS}}(\boldsymbol{\chi}) \geqslant \alpha \right\} \end{array} \right. \tag{4.14}$$

En ce qui concerne le comportement asymptotique du quantile estimé par échantillonnage préférentiel, il est donné par le théorème suivant, issu de [Glynn, 1996] :

Théorème 4.1. *Si* $\mathbb{E}[L(\tilde{\boldsymbol{\xi}}^1)^3] < \infty$ *et la distribution de la sortie* H *est différentiable en* $q_\alpha = H^{-1}(\alpha)$ *avec* $H'(q_\alpha) = h_Y(q_\alpha) > 0$*, alors :*

$$\sqrt{N_{IS}} \left(\tilde{q}_\alpha^{IS} - q_\alpha \right) \underset{N_{IS} \infty}{\xrightarrow{\mathcal{L}}} \sigma_{q,IS} \, \mathcal{N}(0,1) \text{ avec } \sigma_{q,IS}^2 = \frac{\mathrm{Var}\left(L(\tilde{\boldsymbol{\xi}}^1) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha} \right)}{h_Y(q_\alpha)^2} \tag{4.15}$$

Preuve.
Montrons que $\forall N_{IS} \in \mathbb{N}^\star, \quad \tilde{q}_\alpha^{IS} \leqslant t \Leftrightarrow \tilde{H}_{N_{IS}}(t) \geqslant \alpha$. Prenons t tel que $\tilde{q}_\alpha^{IS} \leqslant t$. Alors par définition de $\tilde{q}_\alpha^{IS}$ (étant l'inf de l'ensemble $\left\{ t \ : \ \tilde{H}_{N_{IS}}(t) \geqslant \alpha \right\}$), on a $\tilde{H}_{N_{IS}}(t) \geqslant \alpha$. Dans l'autre sens, si t est tel que $\tilde{H}_{N_{IS}}(t) \geqslant \alpha$, alors par définition de l'inf, on a encore $\tilde{q}_\alpha^{IS} \leqslant t$. Ce qui montre que ces deux inégalités sont équivalentes.

On peut donc écrire :

$$\tilde{\mathbb{P}} \left[\sqrt{N_{IS}} (\tilde{q}_\alpha^{IS} - q_\alpha) \leqslant t \right] = \tilde{\mathbb{P}} \left[\tilde{q}_\alpha^{IS} \leqslant q_\alpha + \frac{t}{\sqrt{N_{IS}}} \right] = \tilde{\mathbb{P}} \left[\alpha \leqslant \tilde{H}_{N_{IS}} \left(q_\alpha + \frac{t}{\sqrt{N_{IS}}} \right) \right] \tag{4.16}$$

Introduisons les notations suivantes

$$\left\{ \begin{array}{rcl} s_{N_{IS}}(t) & = & \left\{ \tilde{\mathbb{E}} \left[L(\tilde{\boldsymbol{\xi}}^1)^2 \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right] - H^2(q_\alpha + t/\sqrt{N_{IS}}) \right\}^{1/2} \\ \psi_{N_{IS}}(t) & = & \dfrac{\sqrt{N_{IS}}}{s_{N_{IS}}(t)} \left[\tilde{H}_{N_{IS}} \left(q_\alpha + \dfrac{t}{\sqrt{N_{IS}}} \right) - H \left(q_\alpha + \dfrac{t}{\sqrt{N_{IS}}} \right) \right] \\ a_{N_{IS}}(t) & = & \dfrac{\sqrt{N_{IS}}}{s_{N_{IS}}(t)} \left[\alpha - H \left(q_\alpha + \dfrac{t}{\sqrt{N_{IS}}} \right) \right] \end{array} \right.$$

Avec ces notations, on a donc :

$$\tilde{\mathbb{P}} \left[\alpha \leqslant \tilde{H}_{N_{IS}} \left(q_\alpha + \frac{t}{\sqrt{N_{IS}}} \right) \right] = \tilde{\mathbb{P}} \left[a_{N_{IS}}(t) \leqslant \psi_{N_{IS}}(t) \right] \underbrace{- \tilde{\mathbb{P}} \left[\mathcal{N}(0,1) \geqslant a_{N_{IS}}(t) \right] + \tilde{\mathbb{P}} \left[\mathcal{N}(0,1) \geqslant a_{N_{IS}}(t) \right]}_{=0!}$$

On peut ainsi réécrire $\psi_{N_{IS}}$ de la manière suivante :

$$\psi_{N_{IS}}(t) = \frac{\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i) \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \leqslant q_\alpha + t/\sqrt{N_{IS}}} - \tilde{\mathbb{E}} \left[L(\tilde{\boldsymbol{\xi}}^1) \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right]}{\sqrt{N_{IS}} \ \left(\mathrm{Var} \left[L(\tilde{\boldsymbol{\xi}}^1) \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right] \right)^{1/2}}$$

$$\text{Car} \left\{ \begin{array}{l} \tilde{H}_{N_{IS}} \left(q_\alpha + \dfrac{t}{\sqrt{N_{IS}}} \right) = \dfrac{1}{N_{IS}} \displaystyle\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i) \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \\ H(q_\alpha + t/\sqrt{N_{IS}}) = \tilde{\mathbb{E}} \left[L(\tilde{\boldsymbol{\xi}}^1) \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right] \\ s_{N_{IS}}^2(t) = \mathrm{Var} \left[L(\tilde{\boldsymbol{\xi}}^1) \ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right] \end{array} \right.$$

En appliquant le théorème de Berry-Esseen [Berry, 1941], on a donc l'inégalité suivante :

$$\sup_x \left| \tilde{\mathbb{P}} \left[\psi_{N_{IS}}(t) \leqslant x \right] - \Phi_{(0,1)}(x) \right| \leqslant C \frac{\tilde{\rho}}{\tilde{\sigma}^3 \sqrt{N_{IS}}}$$

Ici, $\Phi_{(0,1)}$ correspond à la fonction de répartition de la loi normale (définie à l'annexe A), $\tilde{\sigma} = \tilde{\mathbb{E}} \left(\left[L(\tilde{\boldsymbol{\xi}}^1)\ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right]^2 \right)$ et
$\tilde{\rho} = \tilde{\mathbb{E}} \left(\left[L(\tilde{\boldsymbol{\xi}}^1)\ \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha + t/\sqrt{N_{IS}}} \right]^3 \right)$.
Or lorsque $N_{IS} \to \infty$, $\tilde{\rho}$ et $\tilde{\sigma}$ tendent vers une limite finie, grâce aux hypothèses de régularité sur $L(\tilde{\boldsymbol{\xi}}^1)$ du théorème. Ce qui signifie que $C \frac{\tilde{\rho}}{\tilde{\sigma}^3 \sqrt{N_{IS}}} \underset{N_{IS} \infty}{\to} 0$. En repartant de l'équation (4.16), on a donc :

$$\tilde{\mathbb{P}} \left(\sqrt{N_{IS}} (\tilde{q}_\alpha^{IS} - q_\alpha) \leqslant t \right) \underset{N_{IS} \infty}{=} \Phi_{(0,1)} \left(a_{N_{IS}}(t) \right)$$

Et pour le comportement de $a_{N_{IS}}(t)$ quand $N_{IS} \to \infty$, on peut écrire :

$$\begin{aligned} a_{N_{IS}}(t) &= \frac{\sqrt{N_{IS}}}{s_{N_{IS}}(t)} \left[\alpha - H \left(q_\alpha + \frac{t}{\sqrt{N_{IS}}} \right) \right] \\ &= \frac{t}{s_{N_{IS}}(t)} \frac{H(q_\alpha) - H\left(q_\alpha + \frac{t}{\sqrt{N_{IS}}} \right)}{\frac{t}{\sqrt{N_{IS}}}} \\ &\underset{N_{IS} \to \infty}{=} -\frac{t}{s} H'(q_\alpha) = -\frac{t}{\sigma_{q,IS}} \end{aligned}$$

Ce qui nous donne en revenant à l'équation (4.16) pour N_{IS} tendant vers ∞ :

$$\begin{aligned} \tilde{\mathbb{P}} \left(\sqrt{N_{IS}} (\tilde{q}_\alpha^{IS} - q_\alpha) \leqslant t \right) &= \tilde{\mathbb{P}} \left(\mathcal{N}(0,1) \geqslant -\frac{t}{\sigma_{q,IS}} \right) + O\left(\frac{1}{\sqrt{N_{IS}}} \right) \\ &\underset{\times -\sigma_{q,IS}}{=} \tilde{\mathbb{P}} \left(\sigma_{q,IS} \mathcal{N}(0,1) \leqslant t \right) + O\left(\frac{1}{\sqrt{N_{IS}}} \right) \end{aligned} \tag{4.17}$$

□

Sans hypothèses particulières sur la loi biaisée h_b, il est difficile de tirer des conclusions sur la valeur de cette variance asymptotique théorique. En effet, l'équation (4.15) montre que l'erreur d'estimation peut être plus grande que dans le cas d'un Monte-Carlo classique si la densité biaisée est mal choisie ou inversement qu'elle peut diminuer la variance de l'estimation si elle est bien choisie. Étant donné que cette variance est paramétrée par la densité biaisée, il est possible de calculer la densité optimale théorique annulant cette variance. Pour cela, il faut et il suffit de trouver la loi biaisée $h_b^\star$ annulant le numérateur de $\sigma_{q,IS}$:

$$\begin{aligned} &\mathrm{Var} \left(L(\tilde{\boldsymbol{\xi}}^1) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^1) \leqslant q_\alpha} \right) = 0 \\ \Leftrightarrow \quad &\int_{\boldsymbol{\xi} \in \mathcal{D}_\xi} \left(\frac{h(\boldsymbol{\xi})}{h_b^\star(\boldsymbol{\xi})} \mathbf{1}_{\phi(\boldsymbol{\xi}) \leqslant q_\alpha} - \alpha \right)^2 h_b^\star(\boldsymbol{\xi}) d\boldsymbol{\xi} = 0 \end{aligned}$$

Or, si l'intégrale d'une fonction positive est nulle, alors la fonction est nulle presque sûrement. Ceci donne pour la densité optimale $h_b^\star$:

$$h_b^\star(\boldsymbol{\xi}) = \frac{\mathbf{1}_{\phi(\boldsymbol{\xi}) \leqslant q_\alpha}}{\alpha} h(\boldsymbol{\xi}) \tag{4.18}$$

Cette loi biaisée est inconnue puisqu'elle dépend de la quantité recherchée q_α. Toutefois, il existe des algorithmes itératifs permettant d'approcher cette loi optimale. Une première méthode consiste à estimer cette loi optimale sur une famille paramétrique h_λ, en minimisant la divergence de Kullback-Leibler avec la loi optimale. Cette technique s'appelle *Cross-Entropy algorithm* et est présentée en section 4.2.2.3. Une autre méthode permettant d'approcher cette loi biaisée de manière non-paramétrique, à l'aide d'estimateurs à noyaux, y est également détaillée.

Estimateur du superquantile par échantillonnage préférentiel : En repartant de la formulation du superquantile donné à l'équation (4.6), un estimateur par échantillonnage préférentiel peut être proposé. Ce dernier repose sur le tirage de N_{IS} points suivant la densité biaisée h_b, $\left(\tilde{\boldsymbol{\xi}}^1, \cdots, \tilde{\boldsymbol{\xi}}^{N_{IS}}\right)$:

$$\tilde{Q}_\alpha^{IS} = \inf_{\gamma \in \mathbb{R}} \Psi^{N_{IS}}(\gamma) \tag{4.19}$$

$$\text{avec } \Psi^{N_{IS}}(\gamma) = \left[\gamma + \frac{1}{(1-\alpha)\, N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \geqslant \gamma} \left(\phi(\tilde{\boldsymbol{\xi}}^i) - \gamma\right)\right]$$

La variance de cet estimateur a été étudié dans [Hong *et al.*, 2014]. Cette étude a en particulier caractérisé le comportement asymptotique de l'erreur d'estimation par le théorème suivant :

Théorème 4.2. *Si* $\mathbb{E}[\left(\phi(\tilde{\boldsymbol{\xi}}^1) - \gamma\right)^2 L(\tilde{\boldsymbol{\xi}}^1)^2] < \infty$*, alors*

$$\sqrt{N_{IS}}\left(\tilde{Q}_\alpha^{IS} - Q_\alpha\right) \xrightarrow[N_{IS}\infty]{\mathcal{L}} \sigma_{Q,IS}\mathcal{N}(0,1) \text{ avec } \sigma_{Q,IS} = \frac{\sqrt{\text{Var}\left(L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)\geqslant q_\alpha}\left(\phi(\tilde{\boldsymbol{\xi}}^i) - q_\alpha\right)\right)}}{(1-\alpha)} \tag{4.20}$$

Preuve.

La preuve est simple et repose sur le théorème de la limite centrale. Posons $U^i = \frac{\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)\geqslant\gamma}\left(\phi(\tilde{\boldsymbol{\xi}}^i)-\gamma\right)L(\tilde{\boldsymbol{\xi}}^i)}{1-\alpha}$. Alors $U_1, \cdots, U_{N_{IS}}$ est une séquence de variables i.i.d de variance σ^2. On a donc :

$$\left(\Psi^{N_{IS}}(\gamma) - \Psi(\gamma)\right) = \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} U^i - \underbrace{\frac{1}{1-\alpha}\int_{t\in\mathbb{R}} \mathbf{1}_{\phi(t)\geqslant\gamma}\left(\phi(t) - \gamma\right) L(t)}_{\mathbb{E}(U^1)}$$

Par application directe du Théorème de la limite centrale (d'après [Feller, 1966] p259) sur la variable U^i, on a donc :

$$\sqrt{N_{IS}}\left(\Psi^{N_{IS}}(\gamma) - \Psi(\gamma)\right) \xrightarrow[N_{IS}\infty]{\mathcal{L}} \frac{\sqrt{\text{Var}\left(L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)\geqslant\gamma}\left(\phi(\tilde{\boldsymbol{\xi}}^i) - \gamma\right)\right)}}{(1-\alpha)}\mathcal{N}(0,1)$$

Ceci est vrai $\forall\gamma \in \mathbb{R}$ et donc en particulier lorsque $\gamma = q_\alpha$, ce qui permet de montrer le résultat annoncé.

□

Toutefois, il n'existe pas à notre connaissance de travaux étudiant la densité optimale dans le cas du superquantile. Dans son étude [Hong *et al.*, 2014], Hong minimise la variance de l'estimateur en recherchant la loi biaisée par changement de mesure exponentielle aussi appelée *exponential twisting* en anglais. Mais cette technique, notamment détaillée dans [Morio et Balesdent, 2015, pp.57-60], n'est adaptée qu'au cas où la densité de $\phi(\boldsymbol{\xi})$ est connue ou si la fonction $\boldsymbol{\phi}$ dépend linéairement des entrées $\boldsymbol{\xi}$ [Morio et Balesdent, 2015, p.60].

Remarque sur le théorème 4.2 : ce théorème suppose que l'on connaît exactement le quantile q_α, ce qui explique la simplicité de la démonstration. Or, en pratique, cette quantité est trouvée en résolvant le minimum de l'équation (4.19). La variance théorique obtenue dans ce théorème n'est donc pas complète. À la section 8.2.1.2, la variance totale d'estimation du superquantile est étudiée et majorée par la somme de celle obtenue sur le quantile au théorème 4.1 et celle obtenue pour le superquantile au théorème 4.2.

4.2.2.3 Recherche séquentielle de la loi biaisée optimale pour l'estimation de quantile

Pour estimer le quantile par échantillonnage préférentiel, des études approchent la loi biaisée optimale $h_b^\star$ séquentiellement. En effet, étant donné que la loi optimale dépend de paramètres inconnus, tels que la valeur réelle du quantile recherché q_α, il est nécessaire de l'approcher de manière itérative. Ceci se fait à l'aide de nouveaux échantillons de points à chaque itération k $\left(\tilde{\boldsymbol{\xi}}^{k,1}, \cdots, \tilde{\boldsymbol{\xi}}^{k,N_{CE}}\right)$ générés à l'aide d'une loi biaisée qui s'affine avec les itérations. Cette idée provient de l'algorithme de l'entropie croisée issu de [Rubinstein et Kroese, 2004]. L'algorithme 4.2 détaille sa mise en place afin d'estimer un quantile. On peut remarquer que cet algorithme reprend l'idée de l'algorithme 4.1 d'échantillonnage stratifié, en utilisant des β-quantiles empiriques $S_0 < S_1 < \cdots < S_{final} \leqslant \tilde{q}_\alpha^{IS}$ intermédiaires afin d'estimer l'α-quantile final $\tilde{q}_\alpha^{IS}$.

Afin de calculer les paramètres optimaux pour la densité biaisée, l'idée de cet algorithme est de rechercher le $\boldsymbol{\lambda}$ qui fournit une loi $h_{\boldsymbol{\lambda}}(\cdot)$ la plus proche de la loi optimale recherchée $h_b^\star$ de l'équation (4.18) à chaque itération. Pour ce faire, il faut définir la distance entre deux lois : une manière courante est d'utiliser la divergence de Kullback-Leibler [Joyce, 2011]. Elle provient de la théorie de l'information où elle mesure la dissimilarité entre deux densités. Afin de minimiser la dissimilarité entre la loi paramétrique $h_{\boldsymbol{\lambda}}(\cdot)$ et la loi optimale $h_b^\star$, il faut donc minimiser la divergence de Kullback-Leibler D :

$$\begin{aligned} \boldsymbol{\lambda}^\star &= \underset{\boldsymbol{\lambda}}{\operatorname{argmin}}\ D(h_b^\star, h_{\boldsymbol{\lambda}}) \\ &= \underset{\lambda}{\operatorname{argmin}}\ \left[\int_{\mathbb{R}^{N_\xi}} h_b^\star(\boldsymbol{\chi}) \ln\left[h_b^\star(\boldsymbol{\chi})\right] d\boldsymbol{\chi} - \int_{\mathbb{R}^{N_\xi}} h_b^\star(\boldsymbol{\chi}) \ln\left[h_{\boldsymbol{\lambda}}(\boldsymbol{\chi})\right] d\boldsymbol{\chi}\right] \end{aligned} \tag{4.21}$$

Les termes ne dépendant pas de $\boldsymbol{\lambda}$ dans l'équation (4.21) n'influent pas sur les solutions du problème d'optimisation à résoudre, on peut donc retirer ces termes constants par rapport à $\boldsymbol{\lambda}$, ce qui donne en développant $h_b^\star$ d'après l'équation (4.18) :

$$\boldsymbol{\lambda}^\star = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \int_{\mathbb{R}^{N_\xi}} \frac{\mathbf{1}_{\phi(\boldsymbol{\chi})>q_\alpha} \ln\left[h_{\boldsymbol{\lambda}}(\boldsymbol{\chi})\right]}{1-\alpha} h(\boldsymbol{\chi}) d\boldsymbol{\chi}$$

Algorithme 4.2 Algorithme de l'entropie croisée pour l'estimation d'un α-quantile

Input : la densité jointe des entrées h, la fonction $\phi(\cdot)$, le nombre d'échantillons N_{CE} par itération, la valeur des quantiles intermédiaires $\beta \in [0,1]$

Output : le quantile $\tilde{q}_\alpha^{IS}$

· Générer les échantillons i.i.d $\tilde{\boldsymbol{\xi}}^{0,1}, \cdots, \tilde{\boldsymbol{\xi}}^{0,N_{CE}}$ avec la densité h;
· Calculer la valeur des sorties en ces points :
$Y^{0,1} = \phi(\tilde{\boldsymbol{\xi}}^{0,1}), \cdots, Y^{0,N_{CE}} = \phi(\tilde{\boldsymbol{\xi}}^{0,N_{CE}})$;
· Calculer le β-quantile empirique S_0 des $\left(Y^{0,i}\right)_{1\leqslant i\leqslant N_{CE}}$ par l'équation (4.4);
· Calculer l'estimation courante du α-quantile $\tilde{q}_\alpha^{IS,0}$ par 4.14 à partir des $\left(Y^{0,i}\right)_{1\leqslant i\leqslant N_{CE}}$;
· $k = 1$;
while $S_k < \tilde{q}_\alpha^{IS,k}$ **do**
· Optimiser les paramètres de la densité biaisée $\boldsymbol{\lambda}^k$ selon l'équation (adaptée de l'équation (4.23)) :

$$\boldsymbol{\lambda}^k = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^{k-1,i})>S_{k-1}} \frac{h(\tilde{\boldsymbol{\xi}}^{k-1,i})}{h_{\boldsymbol{\lambda}^{k-1}}(\tilde{\boldsymbol{\xi}}^{k-1,i})} \ln(h_{\boldsymbol{\lambda}}(\tilde{\boldsymbol{\xi}}^{k-1,i}))$$

· Générer les échantillons i.i.d $\tilde{\boldsymbol{\xi}}^{k,1} \cdots, \tilde{\boldsymbol{\xi}}^{k,N_{CE}}$ avec la densité $h_{\boldsymbol{\lambda}^k}$;
· Calculer la valeur des sorties en ces points :
$Y^{k,1} = \phi(\tilde{\boldsymbol{\xi}}^{k,1}), \cdots, Y^{k,N_{CE}} = \phi(\tilde{\boldsymbol{\xi}}^{k,N_{CE}})$;
· Calculer le β-quantile empirique S_k des $\left(Y^{k,i}\right)_{1\leqslant i\leqslant N_{CE}}$ par l'équation (4.4);
· Calculer l'estimation courante du α-quantile $\tilde{q}_\alpha^{IS,k}$ par 4.14 à partir des $\left(Y^{k,i}\right)_{1\leqslant i\leqslant N_{CE}}$;
· $k = k + 1$
end
· Optimiser les paramètres de la densité biaisée $\boldsymbol{\lambda}^\star$ selon l'équation (4.23) avec $\tilde{q}_\alpha^{IS,k-1}$ à la place de q_α;
· Générer les échantillons i.i.d $\tilde{\boldsymbol{\xi}}^1, \cdots, \tilde{\boldsymbol{\xi}}^{N_{CE}}$ avec la densité $h_{\boldsymbol{\lambda}^\star}$;
· Estimer le quantile avec le dernier échantillon généré par 4.14;

Une fois encore en retirant les constantes et en se ramenant à une écriture probabiliste, on peut donc écrire :

$$\boldsymbol{\lambda}^\star = \underset{\lambda}{\text{argmax}}\ \mathbb{E}\left(\mathbf{1}_{\phi(\boldsymbol{\xi})>S} \ln(h_{\boldsymbol{\lambda}}(\boldsymbol{\xi})\right) \tag{4.22}$$

Avec l'estimateur par échantillonnage préférentiel, on a donc :

$$\boldsymbol{\lambda}^\star = \underset{\boldsymbol{\lambda}}{\text{argmax}}\ \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} L(\tilde{\boldsymbol{\xi}}^i) \ln(h_{\boldsymbol{\lambda}}(\tilde{\boldsymbol{\xi}}^i)) \tag{4.23}$$

De plus, dans le cas de lois normales pour chaque composante $j \in \{1, \cdots, N_\xi\}$ (voir annexe A), la résolution de ce problème d'optimisation est analytique. En annulant le gradient de la fonction à maximiser dans l'équation (4.23), on obtient deux équations à deux inconnues dont la résolution peut se faire analytiquement et on obtient :

$$\begin{cases} \lambda^\star_{j,1} &= \dfrac{\sum_{i=1}^N L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} \xi^i_j}{\sum_{i=1}^N \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} L(\tilde{\boldsymbol{\xi}}^i)} \\ \lambda^\star_{j,2} &= \dfrac{\sum_{i=1}^N L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} (\xi^i_j - \lambda^\star_{j,1})^2}{\sum_{i=1}^N \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} L(\tilde{\boldsymbol{\xi}}^i)} \end{cases} \tag{4.24}$$

avec $\lambda_{j,1}$ qui correspond à la moyenne de la composante j de la loi biaisée et $\lambda_{j,2}$ qui correspond à la variance de la composante j de la loi biaisée. On peut d'ailleurs remarquer que ces valeurs optimales correspondent exactement à la moyenne pondérée des points $\tilde{\boldsymbol{\xi}}^i$ se situant au dessus du seuil.

Une alternative non-paramétrique pour approximer la loi optimale : alors que l'algorithme par entropie croisée nécessite de choisir une famille paramétrique, il est également possible de ne pas faire ce genre d'hypothèse en approchant la densité optimale de manière non-paramétrique à l'aide d'estimateurs à noyaux. Cette méthode est détaillée dans [Morio, 2012] et [Morio et Balesdent, 2015] et s'appelle NAIS pour *Non-parametric Adaptive Importance Sampling*. L'algorithme à mettre en place est en fait très similaire à l'algorithme 4.2. L'unique différence repose dans le calcul de la loi biaisée. Dans le cas non-paramétrique, la densité biaisée à chaque itération h_k est estimée de la manière suivante :

$$\begin{cases} h_{k+1}(\boldsymbol{\xi}) &= \dfrac{1}{kN_{IS}I_k \det\left(\boldsymbol{B}_k^{1/2}\right)} \displaystyle\sum_{j=1}^{k}\sum_{i=1}^{N_{IS}} \frac{h(\tilde{\boldsymbol{\xi}}^{j,i})}{h_j(\tilde{\boldsymbol{\xi}}^{j,i})} \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^{j,i})>q_\alpha} K\left(\boldsymbol{B}_k^{1/2}\left(\boldsymbol{\xi} - \tilde{\boldsymbol{\xi}}^{j,i}\right)\right) \\ I_k &= \dfrac{1}{kN_{IS}} \displaystyle\sum_{j=1}^{k}\sum_{i=1}^{N_{IS}} \frac{h(\tilde{\boldsymbol{\xi}}^{j,i})}{h_j(\tilde{\boldsymbol{\xi}}^{j,i})} \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^{j,i})>q_\alpha} \end{cases} \tag{4.25}$$

Avec K un noyau non négatif, symétrique et dont l'intégrale vaut 1 et $\boldsymbol{B}_k$ est la matrice des fenêtres (pour *bandwidth* en anglais). Le noyau K permet en fait un lissage de la densité estimée par un Monte-Carlo classique afin d'avoir une approximation plus fine que par une fonction en escalier. Deux exemples de noyau utilisés sont le noyau d'Epanechnikov et le noyau gaussien, qui est en fait la densité centrée d'une loi normale multivariée :

$$K(\boldsymbol{\chi}) = \frac{1}{(2\pi)^{N_\xi/2}} \exp\left(-\frac{1}{2}\boldsymbol{\chi}^T\boldsymbol{\chi}\right)$$

Beaucoup d'études ont montré que le choix du noyau était secondaire par rapport au choix de la fenêtre [Turlach, 1993]. Ce dernier paramètre est délicat à gérer et peut être coûteux à estimer. Une manière de le faire est d'utiliser la validation croisée [Bowman, 1984] (cf section 3.1.2). Si l'utilisateur n'est pas prêt à mettre en place des simulations afin de déterminer ce paramètre, il existe des lois empiriques permettant de choisir une fenêtre avec un ordre de grandeur acceptable. On peut par exemple citer celles proposées par Silverman [Silverman, 1986] ou par Scott [Scott, 1992]. Elles donnent toutes deux une fenêtre proportionnelle à $N_{IS}^{-\frac{1}{5}}$.

4.2.3 Assistance par krigeage

Dans cette partie le modèle de substitution s'écrit $\hat{\phi}$ et on omet l'écriture des paramètres $\boldsymbol{w}$ grâce auxquels ces modèles sont construits.

Afin de réduire le nombre d'appels à la fonction ϕ nécessaires à l'estimation d'un quantile, de nombreuses études ont proposé d'utiliser un modèle de substitution. Pour rappel, un modèle de substitution $\hat{\phi}$ est une fonction analytique construite à partir de simulations provenant de la fonction à substituer. Cet ensemble de simulations s'appelle le plan d'expériences. Un modèle de substitution a donc comme avantage d'apporter une information même sur les points où il n'a pas été calculé, c'est-à-dire hors des points du plan d'expériences. De ce fait, l'utilisation de métamodèles permet de ne pas avoir à estimer tous les points de l'échantillon Monte-Carlo.

Comme détaillé dans la partie 3.2.4, la construction d'un DOE peut en particulier se faire de manière séquentielle, en partant d'échantillons tirés par un LHS d'une taille réduite par rapport au budget total, comme détaillé à l'algorithme 3.5. Alors que dans le cas de la partie 3.2.4, c'est l'erreur globale de la prédiction que l'on souhaite diminuer, dans le cadre de l'estimation de mesures de robustesse, l'erreur globale du modèle de substitution n'est pas importante. En effet, certaines zones n'ont pas besoin d'être connues avec précision et il est possible de cibler les points de l'échantillon Monte-Carlo à rajouter au plan d'expériences qui apportent le plus d'information à l'estimation du quantile. De fait dans le cas du quantile, les algorithmes et les estimateurs présentés ne nécessitent pas de connaître la valeur de la fonction mais uniquement leur position relative au seuil de quantile à estimer. Ceci est illustré sur la figure 4.5 où même si l'erreur d'estimation de certains points peut être grande, il n'est pas nécessaire d'affiner la connaissance de cette région étant donné qu'elle n'apporte pas dans l'immédiat plus d'information sur la valeur du quantile courant. Il n'est donc pas nécessaire de connaître avec précision la valeur de la fonction en tout point, mais il faut uniquement pouvoir assurer que les points de l'échantillon sont au dessus ou en dessous du quantile courant.

En se basant sur les intervalles de confiance fournis par le krigeage, c'est donc uniquement dans l'ensemble de points suivant que le point amenant le plus d'information doit être calculé [Balesdent *et al.*, 2013] :

$$U_{k,S} = \left\{ \boldsymbol{\xi} \in \left(\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^{N_{MC}} \right) \ : \ \hat{\phi}(\boldsymbol{\xi}) - k\hat{\sigma}(\boldsymbol{\xi}) < S < \hat{\phi}(\boldsymbol{\xi}) + k\hat{\sigma}(\boldsymbol{\xi}) \right\} \tag{4.26}$$

Cet ensemble de point est illustré en orange en figure 4.5.

À présent, afin de rajouter au DOE le point qui diminue le plus l'incertitude sur les points de cet ensemble $U_{k,S}$, il est possible d'utiliser les critères d'information introduits dans la partie 3.2.4. Notamment, Balesdent reprend l'idée de la maximisation du critère

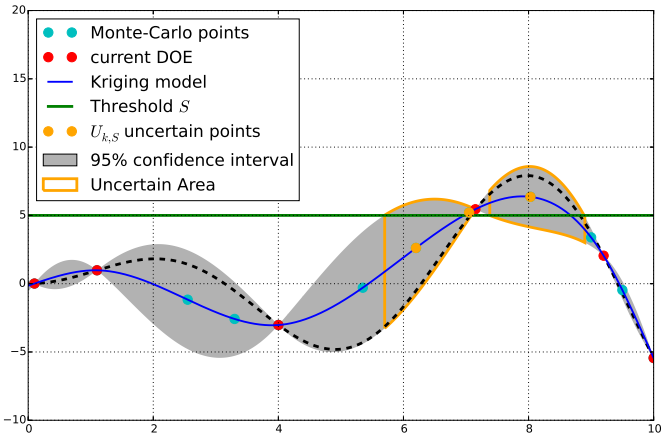


FIGURE 4.5 – Illustration de la zone incertaine $U_{k,S}$ dans le cas d'un quantile ou d'une probabilité de dépassement de seuil S estimée par krigeage

IMSE de l'équation (3.25) dans le papier [Balesdent *et al.*, 2013]. Plus précisément, il discrétise l'intégrale de l'équation (3.25) à l'aide de l'échantillon Monte-Carlo en recentrant chaque terme de la somme par la valeur de la variance $\hat{\sigma}(\boldsymbol{\xi}^i)$ avant l'ajout du point $\boldsymbol{\xi}$, qui est constante par rapport à $\boldsymbol{\xi}$.

$$\mathcal{I}\text{nfo}(\boldsymbol{\xi}) = \text{AN}(\boldsymbol{\xi}) = \sum_{\boldsymbol{\xi}^i \in U_{k,S}} \left[\hat{\sigma}(\boldsymbol{\xi}^i) - \hat{\sigma}^{+\xi}(\boldsymbol{\xi}^i)\right] \tag{4.27}$$

Pour résumer, à chaque fois qu'un échantillon doit être calculé sur la fonction ϕ dans les algorithmes présentés précédemment (4.2 et 4.1), l'algorithme 4.3 doit être appliqué afin d'utiliser le modèle de substitution à la place du modèle réel ϕ et d'enrichir le DOE.

Autres critères d'enrichissement Dans l'algorithme 4.3 l'étape du choix du point $\boldsymbol{\xi}^\star$ maximisant l'apport d'information $\mathcal{I}\text{nfo}(\boldsymbol{\xi})$ peut se faire de différentes manières. Il existe dans la littérature d'autres critères d'enrichissement possibles. Ces derniers proviennent de l'estimation de probabilités de dépassement de seuil et ne sont pas toujours aisément adaptables au cas du quantile.

Tout d'abord, des critères basés sur les intervalles de confiance existent. Par exemple, Dubourg [Dubourg *et al.*, 2011] utilise ces bornes afin de définir un domaine de marge d'incertitude $\mathbb{M}$ autour du seuil à approcher comme à l'équation (4.26) :

$$\mathbb{M} = \left\{\boldsymbol{\xi} \in \mathcal{D}_\xi \ : \ \hat{\phi}(\boldsymbol{\xi}) - k\hat{\sigma}(\boldsymbol{\xi}) < S < \hat{\phi}(\boldsymbol{\xi}) + k\hat{\sigma}(\boldsymbol{\xi})\right\} \tag{4.28}$$

La différence avec précédemment se situe dans le fait que $U_{k,S}$ est un ensemble de points discrets alors que $\mathbb{M}$ est un sous-espace de $\mathcal{D}_\xi$ continu. Une fois cette marge définie, le critère d'enrichissement proposé est de maximiser la probabilité qu'un point se trouve dans cet ensemble $\mathbb{M}$, donc :

$$\mathcal{I}\text{nfo}(\boldsymbol{\xi}) = \mathbb{P}\left[\boldsymbol{\xi} \in \mathbb{M}\right]$$

qui est connu analytiquement grâce aux paramètres du krigeage construit. L'inconvénient de ce critère est qu'il ne quantifie pas l'effet de l'ajout du point $\boldsymbol{\xi}$ sur l'erreur globale

Algorithme 4.3 Enrichissement séquentiel du modèle de krigeage pour le calcul du quantile

Input : Un DOE initial (généré par LHS par exemple)
un échantillon Monte-Carlo $\left(\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^{N_{MC}}\right)$
La taille de l'intervalle de confiance k (1.96 par défaut)
le niveau α de quantile recherché

Output : la valeur du quantile $\hat{q}_\alpha$

· Construire $\hat{\phi}$ à partir du DOE (avec son estimation d'erreur $\hat{\sigma}$);
· Calculer la valeur prédite par $\hat{\phi}$ sur l'échantillon MC :
$\hat{Y}^1 = \hat{\phi}(\boldsymbol{\xi}^1), \cdots, \hat{Y}^{N_{MC}} = \hat{\phi}(\boldsymbol{\xi}^{N_{MC}})$;
· Calculer l'α-quantile $\hat{q}_\alpha$ sur cet échantillon et le domaine de confiance $U_{k,\hat{q}_\alpha}$;
while card $(U_{k,\hat{q}_\alpha}) > 0$ **do**
· Choisir $\boldsymbol{\xi}^\star = \text{argmax}_{\boldsymbol{\xi} \in U_{k,\hat{q}_\alpha}} \mathcal{I}\text{nfo}(\boldsymbol{\xi})$ d'après équation (4.27);
· $DOE = \{DOE, (\boldsymbol{\xi}^\star, \phi(\boldsymbol{\xi}^\star))\}$;
· Construire $\hat{\phi}$ à partir du DOE (avec son estimation d'erreur $\hat{\sigma}$);
· Calculer la valeur prédite par $\hat{\phi}$ sur l'échantillon MC :
$\hat{Y}^1 = \hat{\phi}(\boldsymbol{\xi}^1), \cdots, \hat{Y}^{N_{MC}} = \hat{\phi}(\boldsymbol{\xi}^{N_{MC}})$;
· Calculer l'α-quantile $\hat{q}_\alpha$ sur cet échantillon et le domaine de confiance $U_{k,\hat{q}_\alpha}$;
end

commise par la fonction, à la différence des critères de type IMSE. Toutefois, il a l'avantage d'être immédiat à calculer.

Par ailleurs, il existe des alternatives à l'utilisation de bornes de confiances du krigeage. En voici une liste non exhaustive :

— Un premier critère alternatif est proposé par Echard dans [Echard *et al.*, 2013] :

$$\mathcal{I}\text{nfo}(\boldsymbol{\xi}) = -\frac{\left|S - \hat{\phi}(\boldsymbol{\xi})\right|}{\hat{\sigma}(\boldsymbol{\xi})} \tag{4.29}$$

Il est utilisé dans le cadre de l'algorithme AK-IS permettant de calculer des probabilités de dépassement de seuil S. Dans le cas de l'estimation d'un quantile, S peut par exemple être la valeur de l'α-quantile à l'itération k comme à l'algorithme 4.2. Si l'on détaille le fonctionnement de ce critère, la maximisation de cette équation permet de détecter les points les plus proches du seuil S et dont la variance est maximale. Il revient en quelque sorte à maximiser le critère de variance de l'estimateur du krigeage de l'équation (3.24) sur l'ensemble $U_{k,S}$, ce qui en fait un critère moins informatif que celui de l'équation (4.27) comme expliqué à la section 3.2.4. Cependant, à la différence du critère de l'équation (4.27), celui-ci a l'avantage d'être instantané à calculer.

— D'autres techniques, appelées **SUR** pour *Stepwise Uncertainty Reduction* minimisent la moyenne, conditionnée par le DOE, de la variance de la mesure d'intérêt $\rho_{\boldsymbol{\xi}}$ que l'on cherche à estimer. Cette idée provient de l'optimisation avec le critère d'amélioration espérée (EI pour *Expected Improvement* voir la partie 5.1.3.1) et est notamment reprise par [Bect *et al.*, 2012] dans le cadre de l'estimation de mesures de robustesse. En maximisant ce critère, on espère ajouter le point qui diminue le plus la variance de l'estimation de la mesure de robustesse par le modèle de krigeage

utilisé. La formule du critère à maximiser issue de cette idée est écrite à l'équation (4.30).

$$\mathcal{I}\text{nfo}(\boldsymbol{\xi}) = -\int_{\mathcal{D}^{N_\xi}} \text{Var}\left(\hat{\rho}_{\boldsymbol{\xi}} \left| DOE^{+\boldsymbol{\xi}} \right.\right) \varphi_{\hat{\phi}(\boldsymbol{\xi}),\hat{\sigma}(\boldsymbol{\xi})}(\boldsymbol{\chi}) d\boldsymbol{\chi} \tag{4.30}$$

avec $\varphi_{\hat{\phi}(\boldsymbol{\xi}),\hat{\sigma}(\boldsymbol{\xi})}(\boldsymbol{\chi})$ la densité d'une loi normale de moyenne $\hat{\phi}(\boldsymbol{\xi})$ et d'écart type $\hat{\sigma}(\boldsymbol{\xi})$.

Comme le critère de l'équation (4.27), le critère SUR vise à quantifier l'effet de l'ajout "fictif" d'un point $\boldsymbol{\xi}$ au DOE sur la précision de l'estimation de la mesure de robustesse. Plus précisément, le critère de l'équation (4.27) vise à réduire l'erreur de prédiction du krigeage sur les points de l'échantillon Monte-Carlo où cette erreur contribue à l'erreur d'estimation de la mesure de robustesse (ces points sont ceux qui constituent l'ensemble $U_{k,S}$, sur lequel on applique le critère IMSE). Le critère SUR a lui aussi pour but de mesurer l'effet de l'ajout $\boldsymbol{\xi}$ sur la variance de l'estimation de la mesure de robustesse $\hat{\rho}_{\boldsymbol{\xi}}$. On remarque donc que le critère de l'équation (4.27) peut être vu comme la traduction, sur un échantillon Monte-Carlo, du critère SUR adapté au calcul du quantile.

Dans le cas général pour l'estimation de ce critère SUR, si la mesure à évaluer $\rho_{\boldsymbol{\xi}} = \mathbb{P}(\phi(\boldsymbol{\xi}) > S)$ est une probabilité de dépassement de seuil S, alors il existe des formulations analytiques de ce critère [Chevalier *et al.*, 2014]. En revanche, dans le cas du quantile $\rho_{\boldsymbol{\xi}} = q_\alpha$, il est nécessaire de passer par des estimations Monte-Carlo de ce critère, ce qui augmente la complexité de son estimation.

4.3 Analyse de sensibilité

L'objectif de l'analyse de sensibilité est de quantifier l'effet d'incertitudes entachant les paramètres d'entrée sur la variabilité de la sortie d'un modèle. Cela peut notamment servir à identifier les entrées les plus influentes, à les hiérarchiser selon leur influence ou encore à déterminer les entrées ayant le moins d'influence de sorte à pouvoir les fixer à une valeur constante. Selon le budget de calcul et la précision recherchée par l'ingénieur, il existe différentes manières de quantifier ces effets. Une revue extensive de ces méthodes est donnée par Iooss [Iooss et Lemaître, 2015].

À l'origine, la sensibilité d'un modèle à ses entrées était déterminée grâce à l'estimation de dérivées partielles de la sortie par rapport à chacune des entrées et ce autour d'un point nominal. En effet, ces dérivées permettaient de donner une information locale sur la dépendance d'une fonction à chacune de ses entrées ξ_i. Toutefois, la localité de l'information peut ne pas être satisfaisante et de plus un gradient n'est pas toujours calculable, notamment dans le cas de l'utilisation de boîtes noires.

Par la suite, afin de quantifier globalement cette sensibilité, de nouvelles techniques basées sur des théories statistiques ont vu le jour. Ces théories permettent en particulier de faire de l'analyse de sensibilité globale, c'est-à-dire prenant en compte l'effet d'une variable sur l'ensemble de son domaine de définition et plus seulement autour d'un état nominal. Il existe dans ce domaine différentes méthodes, chacune ayant un intérêt selon le budget de calcul et les effets recherchés sur la sortie du modèle.

Par exemple dans le cas de la très grande dimension ou d'un budget de calcul réduit, le criblage, aussi appelé *screening* en anglais, peut être une solution. Le criblage consiste à déterminer les effets des entrées et de leurs interactions à partir d'un plan d'expériences. Par exemple, les plans factoriels présentés dans la partie 3.1.3 permettent d'estimer les interactions et leurs effets. Cependant, comme expliqué précédemment, cette méthode devient très rapidement inutilisable dans le cas d'un budget de calcul limité. À l'opposé, le plan d'expériences le plus simple mais ne permettant que d'estimer les effets principaux avec une précision inconnue est le plan OAT pour *One At a Time* [Kleijnen, 2007]. Cette méthode est basée sur la décomposition de chacune des dimensions en niveaux. Elle consiste ensuite à partir d'un point initial et à changer le niveau d'une seule entrée à la fois et en limitant le nombre de niveaux à deux ou trois par dimension. Par exemple dans le cas de seulement deux niveaux, ce plan ne requiert que $N_\xi + 1$ calculs. Entre ces deux plans, il existe une multitude d'alternatives, selon ce que souhaite estimer l'ingénieur et avec quelle précision. L'une des ces alternatives est proposée par Morris [Morris, 1991] et consiste à répéter un certain nombre de fois un plan OAT aléatoirement. Ceci permet notamment de classer les entrées entre : celles qui sont négligeables, celles qui ont des effets linéaires sans interaction et celles qui ont des effets non-linéaires et/ou avec interactions.

Toutefois, ces méthodes permettent d'obtenir des résultats qualitatifs et non quantitatifs. Une technique basée sur l'estimation de la variance de la sortie due à chacune des entrées a récemment fait l'objet d'intenses recherches. Le principe de cette technique est d'abord présenté, puis les principaux axes de recherches présents dans la littérature sont détaillés.

4.3.1 Méthodes basées sur la variance fonctionnelle - Indices de Sobol

Pour simplifier l'écriture des équations, on se place dans le cas où les incertitudes suivent une loi uniforme $\boldsymbol{\xi}_i \sim \mathcal{U}([0,1])$ et sont indépendantes deux à deux. Cette théorie reste valable même dans le cas où ces incertitudes suivent une loi quelconque, à condition que les différentes composantes soient indépendantes deux à deux.

Afin de quantifier la part de variabilité provenant de chacune des entrées, l'idée proposée par Sobol [Sobol′, 2001] est basée sur la décomposition de Hoeffding [Hoeffding, 1948] d'une fonction ϕ en somme de fonctions élémentaires. Sous l'hypothèse que la variance de la sortie est finie, il est possible de décomposer ϕ de la manière suivante :

$$\begin{aligned} \phi(\boldsymbol{\xi}) &= \phi_0 + \sum_{i=1}^{N_\xi} \phi_i(\xi_i) + \cdots + \phi_{1\cdots N_\xi}(\xi_1, \cdots, \xi_{N_\xi}) \\ &= \sum_{\boldsymbol{u} \subseteq \{1,2,\cdots,N_\xi\}} \phi_{\boldsymbol{u}}(\boldsymbol{\xi}_{\boldsymbol{u}}) \end{aligned} \tag{4.31}$$

Avec $\boldsymbol{u}$ étant un sous-ensemble d'indices tiré de l'ensemble $\{1, 2, \cdots, N_\xi\}$. Afin de garantir l'unicité de cette décomposition, il faut imposer un ensemble de conditions sur les fonctions $\phi_{\boldsymbol{u}}$. Ces conditions s'écrivent de la manière suivante :

$$\int_{\xi_i \in [0,1]} \phi_{\boldsymbol{u}}(\xi_{u_1}, \cdots, \xi_{u_{n_u}}) d\xi_i = 0 \quad \begin{cases} \forall i \in \boldsymbol{u} \\ \forall \boldsymbol{u} \subseteq \{1, 2, \cdots, N_\xi\} \\ \text{card}(\boldsymbol{u}) \geqslant 1 \end{cases}$$

Dans le cas où les incertitudes ne suivent pas une loi uniforme, il est également possible de garantir l'unicité en vérifiant des conditions similaires que l'on peut par exemple trouver dans [Chastaing *et al.*, 2012].

À présent, si les composantes du vecteur d'entrées $\boldsymbol{\xi}$ sont indépendantes deux à deux, alors on peut calculer la décomposition de la variance fonctionnelle (aussi appelée décomposition ANOVA) à partir de l'équation (4.31) :

$$\text{Var}[Y] = \sum_{i=1}^{N_\xi} V_i(Y) + \sum_{i<j} V_{ij}(Y) + \cdots + V_{1\cdots N_\xi}(Y) \tag{4.32}$$

$$\text{Avec :} \quad \begin{cases} V_i(Y) = \text{Var}\{\mathbb{E}[\phi(\boldsymbol{\xi}) | \xi_i]\} \\ V_{ij}(Y) = \text{Var}\{\mathbb{E}[\phi(\boldsymbol{\xi}) | \xi_i \xi_j]\} - V_i(Y) - V_j(Y) \\ \cdots \end{cases}$$

On peut donc définir les indices de Sobol liés à chaque entrée ξ_i comme la part due à l'entrée ξ_i de la variance totale de la sortie donc :

$$S_i(Y) = \frac{V_i(Y)}{\text{Var}(Y)}$$

Plus généralement, on a dans le cas de plusieurs indices $\forall \boldsymbol{u} \subseteq \{1, 2, \cdots, N_\xi\}$:

$$S_{\boldsymbol{u}}(Y) = \frac{V_{\boldsymbol{u}}(Y)}{\text{Var}(Y)} \tag{4.33}$$

Dans le cas où les entrées sont indépendantes deux à deux, ces indices ont les propriétés suivantes :

— Ils sont tous compris entre 0 et 1 : $\forall \boldsymbol{u} \subseteq \{1, \cdots N_\xi\}$, $S_{\boldsymbol{u}} \in [0, 1]$
— Grâce à l'équation (4.32), on a de plus :

$$\sum_{\boldsymbol{u} \subseteq \{1, \cdots N_\xi\}} S_{\boldsymbol{u}} = 1$$

Ceci rend leur interprétation aisée, puisque chacun de ces coefficients correspond donc au pourcentage de la variance de la sortie due aux entrées $\boldsymbol{\xi_u} = (\xi_{u_1}, \cdots, \xi_{u_{n_u}})$
— Plus la dimension N_ξ est grande, plus il y a d'indices à calculer et à interpréter. Afin de s'affranchir de ce problème, les indices de Sobol totaux S_{T_i} sont une alternative intéressante puisqu'ils prennent en compte l'intégralité des effets d'une entrée ξ_i sur la variance de la sortie en regroupant les indices partiels :

$$S_{T_i} = S_i + \sum_{i \neq j} S_{ij} + \sum_{j \neq i, k \neq i, j < k} S_{ijk} + \cdots = \sum_{\substack{\boldsymbol{u} \subseteq \{1, \cdots N_\xi\} \\ i \in \boldsymbol{u}}} S_{\boldsymbol{u}} \tag{4.34}$$

De ce fait, il est fréquent de ne calculer que les indices du premier ordre S_i et les indices totaux S_{T_i} pour chaque entrée ξ_i.

4.3.2 Méthodes d'estimation des indices de Sobol

Les manières d'estimer la variance détaillées dans la partie 4.2 restent valables dans cette partie. Toutefois, étant donné le grand nombre de variances à estimer, il existe des algorithmes permettant d'estimer plusieurs d'entre elles avec un même échantillon Monte-Carlo [Saltelli, 2002]. Notamment, la technique du pick-freeze permet efficacement d'estimer les différents indices de Sobol [Gamboa *et al.*, 2015]. Il existe d'autres méthodes d'estimation n'utilisant pas les méthodes de Monte-Carlo. Par exemple la méthode FAST basée sur la transformée de Fourier multidimensionnelle de ϕ est une méthode moins coûteuse que les estimations Monte-Carlo et permet d'estimer les indices principaux et totaux. Toutefois, elle n'est pas adaptée aux cas où la dimension est trop élevée [Tissot et Prieur, 2012].

Sachant que les méthodes précédentes peuvent aisément requérir plus de 1000 appels à la fonction ϕ, elles peuvent donc être inenvisageables dans beaucoup de cas. Afin de proposer des méthodes nécessitant moins d'appels, il est possible d'utiliser des modèles de substitution pour estimer les coefficients de Sobol. Par exemple, les polynômes de chaos permettent de calculer analytiquement les coefficients [Sudret, 2008]. Grâce à cela, le coût d'estimation des coefficients se réduit uniquement au coût de construction du polynôme de chaos. Autre alternative, l'utilisation d'un modèle de krigeage, détaillée en section 3.2.3, permet également d'estimer les coefficients de Sobol de manière analytique. De plus, elle a l'avantage de fournir une mesure de l'incertitude sur l'estimation des coefficients de Sobol due à l'approximation par le métamodèle [Marrel *et al.*, 2009]. Cette connaissance permet notamment d'améliorer cette estimation grâce à un enrichissement séquentiel du plan d'expériences du modèle de substitution avec le point qui diminue le plus l'erreur commise sur l'estimation des coefficients, comme détaillé en section 3.2.4.

Utilisation des DGSM : Par ailleurs, si l'ingénieur ne souhaite qu'éliminer les entrées non influentes alors la connaissance d'une borne supérieure des indices totaux peut suffire. Sous certaines hypothèses, cette borne peut être fournie par l'estimation des coefficients DGSM (pour *Derivative-based Global Sensitivity Measures*) [Sobol et Kucherenko, 2009]

qui représentent donc une alternative intéressante. Néanmoins, ils nécessitent de connaître la dérivée partielle des sorties par rapport aux entrées du modèle. En effet, leur définition est basée sur l'utilisation de ces dérivées dans le but d'estimer l'influence de chaque entrée sur la sortie de la fonction. Or, la dérivée apporte une mesure locale et les DGSM ont pour but de fournir une estimation globale. Pour répondre à cela, l'idée proposée par Kucherenko est de définir le coefficient de sensibilité ν_i de la sortie par rapport à l'entrée ξ_i comme l'intégrale de la dérivée partielle de la sortie par rapport à une entrée sur tout le domaine de variation. Plus précisément, c'est le carré de la dérivée partielle qui doit être intégré et ν_i est définie par l'équation (4.35) :

$$\nu_i = \mathbb{E}\left[\left(\frac{\partial\phi(\boldsymbol{\xi})}{\partial\xi_i}\right)^2\right] = \int_{\boldsymbol{z}\in\mathcal{D}_\xi}\left(\frac{\partial\phi(\boldsymbol{z})}{\partial\xi_i}\right)^2 d\boldsymbol{z} \tag{4.35}$$

Ces coefficients sont bien définis si et seulement si chacune des dérivées partielles de ϕ est de carré intégrable, c'est-à-dire de variance finie.

Pour faire le lien avec les coefficients de Sobol, il faut utiliser l'inégalité de Poincaré. Grâce à elle, il peut être montré que les coefficients DGSM ν_i fournissent une borne supérieure des coefficients de Sobol totaux S_{T_i} dans le cas où les entrées sont indépendantes deux à deux et si ces entrées suivent une loi d'un certain type [Lamboni *et al.*, 2013]. Plus précisément, il faut que la distribution des entrées appartienne à la classe des mesures de Boltzmann, ce qui n'est pas une hypothèse restrictive. Si ces conditions sont vérifiées, on a donc :

$$S_{T_i} \leqslant C(h)\ \nu_i \tag{4.36}$$

avec C qui dépend de la loi des entrées h.

En ce qui concerne l'estimation de ces paramètres, les méthodes de Monte-Carlo sont par exemple toujours utilisables [Kucherenko *et al.*, 2009].

4.3.3 Études récentes élargissant le champ d'application des indices de Sobol

La théorie de base des indices de Sobol repose sur des hypothèses qui ne sont pas toujours vérifiées en pratique. Il est donc intéressant ici de montrer les avancées récentes permettant de dépasser certaines des limites que présente la théorie initiale :

- L'hypothèse principale sur laquelle repose l'interprétabilité des coefficients de Sobol est l'indépendance deux à deux des incertitudes en entrée. Toutefois, dans beaucoup de cas « réels », les incertitudes portant sur différentes entrées peuvent être corrélées. Dans ce cas, les coefficients de Sobol définis comme à l'équation (4.33) ne représentent plus un pourcentage de la part de variabilité de la sortie. Chastaing [Chastaing *et al.*, 2012] a généralisé la décomposition de Hoeffding dans le cas où les corrélations entre deux entrées ont une forme particulière. Cela lui a permis de définir de nouveaux coefficients de sensibilité qui sont une généralisation des coefficients de Sobol précédents. D'autres études, comme celle de Da Veiga [Da Veiga *et al.*, 2009], utilisent des techniques alternatives à FAST ou à Monte-Carlo sans redéfinir les coefficients. Ces alternatives sont indispensables puisque les méthodes FAST et Monte-Carlo modifié ne sont plus utilisables dans le cas corrélé.
- Une autre limite des coefficients de Sobol est qu'ils ne sont pas définis pour prendre en compte le cas d'entrées et sorties fonctionnelles, par exemple dépendantes du

temps. Il existe un certain nombre d'études travaillant sur ce sujet. On peut par exemple citer Fruth [Fruth *et al.*, 2015] qui, dans le cas d'entrées fonctionnelles, se ramène à un cadre classique en traduisant des entrées fonctionnelles en une succession d'approximations linéaires. Grâce à cela, le problème initial, fonctionnel donc en dimension infinie, peut être traité comme un problème de dimension fini et donc adapté au calcul de coefficients de Sobol. Autre exemple, Gamboa [Gamboa *et al.*, 2014] redéfinit quant à lui la décomposition de Hoeffding dans le cas de sorties fonctionnelles. Ceci lui permet d'adapter la définition des coefficients à cette caractéristique et donc de pouvoir estimer la sensibilité de sorties fonctionnelles aux paramètres d'entrée.

— Par définition, les coefficients de Sobol permettent d'estimer l'effet d'incertitudes en entrée sur la variance de la sortie du modèle ϕ. Or, il se peut que l'ingénieur ne soit pas intéressé par l'effet sur la variance, qui n'est pas une mesure de risque satisfaisante. Pour répondre à cela, il existe des études redéfinissant les coefficients de sorte à mesurer l'effet des entrées sur d'autres mesures de robustesse : par exemple Morio [Morio, 2011] s'intéresse à l'effet sur une probabilité de défaillance, Borgonovo [Borgonovo, 2007] a regardé l'effet des paramètres d'entrée sur la distribution de sortie ou encore Fort [Fort *et al.*, 2015] s'est intéressé à l'effet des incertitudes en entrée sur des fonctions de contraste appliquées à la sortie. Ceci inclut l'espérance, la médiane, les quantiles, les probabilités de dépassement de seuil et fournit donc un résultat très général.

Chapitre 5

Optimisation de fonctions boîtes noires

Sommaire

Dans cette partie, nous introduisons le concept d'optimisation et détaillons les manières de résoudre un problème présentant un ou plusieurs objectifs à minimiser, avec ou sans incertitudes. Formellement, on cherche à résoudre le problème multi-objectif suivant :

$$\min_{\boldsymbol{x}\in\mathcal{D}_x} \Big[f_1(\boldsymbol{x}), \cdots, f_{N_{obj}}(\boldsymbol{x})\Big] \tag{5.1}$$

Ici, les fonctions objectif f_i aussi appelées fonctions coût sont définies $\forall i,\ 1 \leqslant i \leqslant N_{obj}$ de la manière suivante :

$$\begin{array}{rccc} f_i & : & \mathcal{D}_x \subset \mathbb{R}^{N_x} & \longrightarrow & \mathbb{R} \\ & & \boldsymbol{x} & \longmapsto & f_i(\boldsymbol{x}) \end{array}$$

$\mathcal{D}_x$ est le domaine de définition des variables d'optimisation $\boldsymbol{x}$, aussi appelées variables de contrôle. Dans cette étude, $\mathcal{D}_x$ est un sous-ensemble borné de $\mathbb{R}^{N_x}$. On se place dans le cadre où les fonctions étudiées sont des boîtes noires (cf figure 3.1) : ceci signifie que l'unique connaissance disponible sur ces fonctions est leur réponse à des entrées données. De ce fait, toutes les techniques basées sur la connaissance ou le calcul du gradient de la

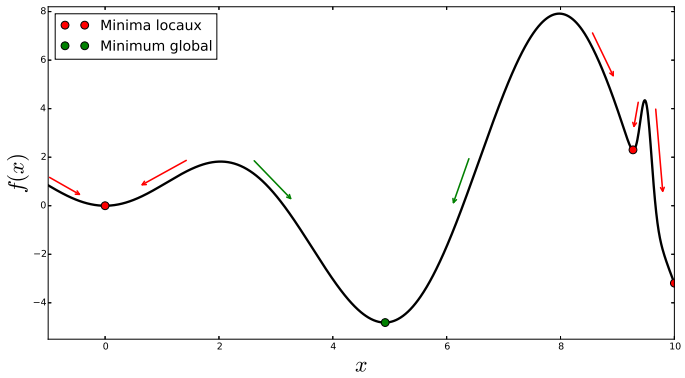


FIGURE 5.1 – Illustration des minima locaux et du minimum global d'une fonction 1D. Les directions fournies par le gradient sont représentées par des flèches.

fonction objectif sont exclues du spectre de notre étude et il n'est donc question que de l'utilisation de techniques sans dérivée que l'on peut trouver dans la littérature sous le nom de *derivative-free optimization*.

La première partie présente les algorithmes sans gradient, avec ou sans modèles de substitution, permettant de résoudre les problèmes du type mono-objectif.

Notre problème industriel nécessite la minimisation simultanée de deux objectifs. Dans la seconde partie nous présentons les différentes façons de prendre en compte plusieurs objectifs dans un problème d'optimisation. Selon ce que souhaite l'utilisateur, il existe différentes façons de formuler le problème. Nous détaillons ensuite les algorithmes sans gradient, avec ou sans modèles de substitution.

Parfois, certaines entrées du problème d'optimisation à résoudre peuvent être incertaines. La dernière partie discute de l'importance de la quantification des incertitudes dès la formulation d'un problème d'optimisation. Pour cela, différentes techniques de la littérature sont présentées. Ces méthodes sont basées sur l'estimation des mesures de robustesse présentées au chapitre précédent.

5.1 Optimisation Mono-objectif

L'optimisation est une branche des mathématiques cherchant à trouver analytiquement ou numériquement le minimum d'une fonction f dépendant de paramètres d'optimisation $\boldsymbol{x}$ qui influent sur la sortie de la fonction $f(\boldsymbol{x})$. Plus précisément, c'est la valeur $\boldsymbol{x}^\star$ où la fonction f est minimum qui est recherchée puisqu'elle correspond aux paramètres optimaux que l'utilisateur recherche. Ces paramètres peuvent appartenir à un domaine à évènements discrets ou continus. Dans le premier cas, on parle d'optimisation combinatoire. Cette étude se focalise sur le deuxième cas, aussi appelé optimisation continue. Les paramètres d'optimisation sur lesquels peut jouer l'utilisateur sont donc de la forme suivante : $\boldsymbol{x} \in \mathbb{R}^{N_x}$ et $f : \mathbb{R}^{N_x} \longrightarrow \mathbb{R}$.

5.1.1 Algorithmes avec gradients

Historiquement, les premières techniques d'optimisation sont celles basées sur l'utilisation des dérivées. Deux exemples de ce type de techniques sont la méthode de Levenberg-Marquardt [Moré, 1978] ou la méthode BFGS [Fletcher, 1987], déjà présentées dans le cas

de l'apprentissage de réseaux de neurones à la section 3.2.2.1. Leur idée est d'itérativement s'approcher du minimum en suivant une direction calculée à partir du gradient, et éventuellement de la Hessienne ou d'une approximation de cette dernière, de la fonction à minimiser. Ces directions sont illustrées par des flèches sur la figure 5.1. Elles nécessitent donc de connaître davantage que la valeur de la fonction à minimiser. De plus, si cette fonction n'est pas convexe, les méthodes de gradient ont pour inconvénient de ne converger que vers des minima locaux dépendants du point de départ, ce qui est également illustré sur la figure 5.1. Ces méthodes peuvent toutefois être utilisées afin de rechercher le minimum global en exécutant cet algorithme à partir de divers points d'initialisation, par exemple tirés aléatoirement. Cette technique est communément appelée *multistart*.

5.1.2 Algorithmes sans gradient

Le gradient et la Hessienne de la fonction à minimiser apportent une information sur sa courbure et donc sur la direction à prendre pour se rapprocher d'un minimum local. Il est nécessaire de compenser leur absence pour pouvoir développer des techniques sans gradient. Une première manière de faire est de choisir la direction de descente à l'aide de la valeur de la fonction f sur un échantillon de points tirés localement, ce qui revient en quelque sorte à approcher le gradient à partir d'une population. La deuxième manière de faire est de mélanger l'exploration du domaine entier et la recherche locale, aussi appelée intensification (ou exploitation). Deux algorithmes de ce type sont présentés : un déterministe appelé DIRECT et un stochastique appelé algorithme à évolution différentielle.

5.1.2.1 Algorithmes de recherche locale

Une première façon de compenser l'absence d'estimateurs du gradient est de s'inspirer des algorithmes basés sur la descente de gradient. Ces dernières partent d'un point initial puis suivent la direction que cette information fournit, c'est-à-dire celle permettant de faire décroître la fonction f. Pour pouvoir approximer la courbure locale de la fonction à optimiser, certains algorithmes sans gradient utilisent un échantillon de points localisés dans une zone réduite du domaine de définition $\mathcal{D}_x$. Ensuite, les points de l'itération courante minimisant la fonction f sont utilisés afin de choisir les points à générer à l'itération suivante. En itérant jusqu'à satisfaire un critère d'arrêt qui peut être propre à la méthode utilisée, la procédure converge vers le minimum le plus proche. Cette trame, détaillée dans l'algorithme 5.1, est la même pour beaucoup d'algorithmes d'optimisation sans gradient. Par exemple, Nelder-Mead ou CMA-ES la respectent. La première a un intérêt historique puisque c'est l'une des premières techniques d'optimisation sans gradient appliquées au cas continu. La deuxième a fait récemment l'objet de nombreuses publications et est un domaine de recherche très actif du fait du nombre réduit de paramètres à gérer et de la robustesse de l'algorithme. Seule l'étape de génération et du choix des nouveaux points à partir de la valeur de f sur la population courante diffère.

Génération des points dans le cas de Nelder-Mead L'algorithme de Nelder-Mead [Nelder et Mead, 1965] est un premier exemple d'algorithmes d'optimisation sans gradient. Il travaille sur une population de N_x+1 points qui forment un simplexe. Ces N_x+1 points se déplacent au cours de l'optimisation en fonction de la valeur prise par f en ces points : en comparant la valeur du minimum à celle du maximum sur la population à chaque itération, le simplexe s'étire, se contracte ou se déplace en allant dans le sens du minimum. Cette

Algorithme 5.1 Algorithme d'optimisation basés sur une recherche locale

Input : La fonction à minimiser f
Le domaine de définition $\mathcal{D}_x$
Nombre de points par génération N_{gen}

Output : Le minimum de f, $f(\boldsymbol{x}^\star)$

· Générer la population initiale dans $\mathcal{D}_x \rightarrow \boldsymbol{X} = \left(\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{N_{gen}}\right)$;
while *Critère d'arrêt non satisfait* **do**
· Calculer la valeur de la fonction aux points de la population $\rightarrow \boldsymbol{Y} = f(\boldsymbol{X})$;
· À partir de $\boldsymbol{Y}$ choisir les $\boldsymbol{x}^i$ minimisant f sur la population courante $\boldsymbol{X}$;
· Générer de nouveaux points à partir de ces $\boldsymbol{x}^i \rightarrow \boldsymbol{X}$;
end
· Retourner min $f(\boldsymbol{X})$;

technique est quelquefois présentée comme une approximation du gradient à l'ordre 0. Par ailleurs, elle présente l'avantage d'avoir peu de paramètres à gérer par l'utilisateur puisqu'il n'y a que le simplexe initial à fixer. Toutefois, cet algorithme a un caractère local et peut manquer de robustesse à proximité des bords du domaine de définition. Il est donc fréquent de l'exécuter avec plusieurs initialisations différentes.

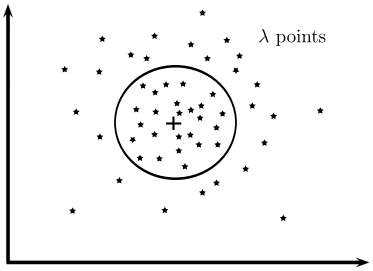
Génération des points dans le cas de CMA-ES CMA-ES est un algorithme évolutionnaire, donc basé sur une population de λ points évoluant avec les itérations. Son principe est par exemple développé par Hansen dans son étude [Hansen, 2006]. CMA-ES signifie *Covariance Matrix Adaptation Evolution Strategy*. À la différence de Nelder-Mead qui est un algorithme déterministe, c'est une méthode stochastique basée sur la génération de points selon une loi normale multivariée à chaque itération. Soit $\boldsymbol{m}$ la moyenne et soit $\boldsymbol{C}$ la matrice de covariance de cette loi multivariée, l'idée proposée par Hansen est donc de générer $\boldsymbol{X}$ selon une loi $\mathcal{N}(\boldsymbol{m}, \boldsymbol{C})$ à chaque itération. Ceci est illustré par la figure 5.2. Afin de d'approcher du minimum, cette moyenne et cette matrice de covariance sont modifiées à l'aide de la valeur de la fonction f sur les μ meilleurs échantillons de la population courante $\boldsymbol{Y} = f(\boldsymbol{X})$. La matrice de covariance permet d'élargir le domaine de recherche ou de le restreindre, selon la valeur des points échantillonnés à l'itération courante. La moyenne permet de déplacer le domaine de recherche vers les points de valeur minimum. Les seuls paramètres à fixer par l'utilisateur sont le nombre λ de points générés à chaque itération, le nombre μ de points à partir desquels choisir les paramètres de l'itération suivante et le poids donné par l'utilisateur à ces μ points dans le calcul de la nouvelle matrice de covariance c_{cov}. Enfin, comme d'autres méthodes locales, il est nécessaire de donner un point d'initialisation pour la moyenne de la loi multivariée initiale.

5.1.2.2 Algorithmes de recherche globale

D'autres techniques échantillonnent de manière déterministe ou stochastique le domaine de définition entier de la fonction. Alors que les techniques échantillonnant localement ont pour but de favoriser l'intensification dans la zone dans laquelle elle est initialisée, les méthodes d'échantillonnage global doivent permettre de faire le compromis entre intensification et exploration. Du fait que ces techniques échantillonnent des domaines plus conséquents que les méthodes locales, le nombre d'appels aux objectifs est donc plus

Étape de générations des entrées $\boldsymbol{X}$

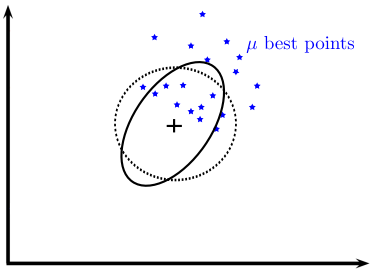
$$\begin{aligned} \boldsymbol{x}^i &= \boldsymbol{m} + \boldsymbol{\eta}^i \\ \boldsymbol{\eta}^i &\sim \mathcal{N}(0, \boldsymbol{C}) \\ i &= 1, \cdots, \lambda \end{aligned}$$



Sélection des μ points de valeur minimum sur f (avec $\eta^{(i)}$ classés croissants selon la valeur de $f(x^{(i)})$)

$$\bar{\boldsymbol{\eta}} = \frac{1}{\mu} \sum_{i=1}^{\mu} \boldsymbol{\eta}^{(i)}$$

$$\boldsymbol{C}^{k+1} \leftarrow (1 - c_{cov})\boldsymbol{C}^k + c_{cov}\mu\bar{\boldsymbol{\eta}}\bar{\boldsymbol{\eta}}^T$$



Calcul du nouveau centre $\boldsymbol{m}$

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \boldsymbol{\eta}^w$$

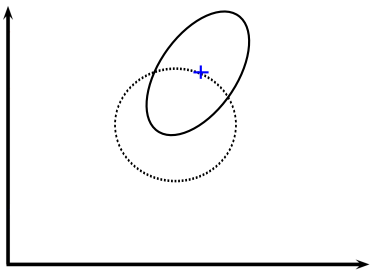


FIGURE 5.2 – Différentes étapes de génération des points dans le cas de l'algorithme CMA-ES

grand. Toutefois, ces techniques de recherche globale ne nécessitent pas de *multistart* pour converger vers le minimum global.

On peut distinguer deux manières différentes d'échantillonner l'espace. La première manière est déterministe et est basée sur le découpage de l'espace de design en mailles (par exemple des hyper-rectangles) et au raffinement de ce découpage suivant la valeur que prend la fonction au centre de ces mailles. Elle a l'avantage d'être applicable à des fonctions objectif de nature très diverses mais a pour désavantage de nécessiter un nombre d'appels très important, ce nombre d'appels augmentant grandement avec le nombre de dimensions.

Une deuxième manière est d'échantillonner l'espace de manière stochastique. Ceci consiste à échantillonner le domaine de définition afin de générer une population que l'on fait ensuite évoluer de manière aléatoire. L'introduction du stochastique dans la recherche du minimum permet notamment d'être plus flexible dans l'exploration. Plus précisément, l'évolution de la population est conditionnée par des opérateurs qui sont définis de sorte à réaliser le compromis exploration/intensification de manière plus malléable au cours de l'algorithme que dans le cas déterministe. Cependant, elles nécessitent donc de fixer un nombre de paramètres plus conséquent et le nombre d'appels nécessaires à la convergence

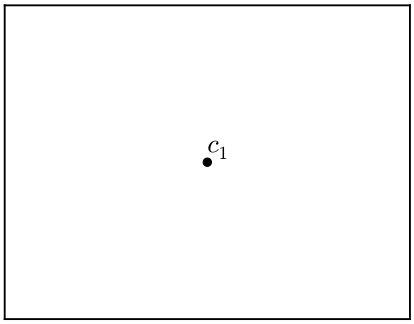
peut être grand. Elles peuvent être rencontrées dans la littérature sont le nom de métaheuristique [Siarry, 2014]. Les algorithmes évolutionnaires sont un exemple de ces techniques et sont détaillés dans la section 5.1.2.4.

5.1.2.3 Un exemple d'algorithme d'optimisation global déterministe : DIRECT

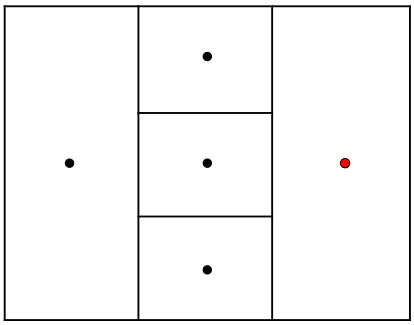
Cet algorithme, initialement proposé par Jones [Jones *et al.*, 1993], a été développé afin de minimiser des fonctions dont le domaine de définition $\mathcal{D}_x$ est borné et dans le cas où la fonction à minimiser n'est pas simple, c'est-à-dire non-linéaire, non convexe et dont le gradient n'est pas connu. Il est basé sur un échantillonnage déterministe de la fonction à minimiser. Pour cela, cette méthode s'inspire de la technique du *branch and bound* provenant de l'optimisation combinatoire. Plus précisément, elle consiste à mailler le domaine de définition entier à l'aide d'hypercubes en calculant la valeur de la fonction f au centre de chacun d'eux et en raffinant les zones où la fonction f est petite. C'est donc un mélange d'exploration et d'intensification. Alors que les algorithmes de la section précédente se focalisent dans une zone du domaine de définition, la recherche globale réalisée par DIRECT porte sur tout le domaine de définition.

Les étapes de l'algorithme DIRECT sont les suivantes :

1. Le domaine de définition $\mathcal{D}_x$ étant borné, il peut être transformé en un hypercube de dimension N_x : $\bar{\mathcal{D}}_x = \left\{ \boldsymbol{x} \in \mathbb{R}^{N_x} \ : \ \forall i \in [\![1, N_x]\!], \ 0 \leqslant x_i \leqslant 1 \right\}$. La valeur de la fonction est évaluée au centre de cet hypercube $\boldsymbol{c}_1$.



2. L'hypercube initial est ensuite divisé en hyper-rectangles à l'aide de la distance entre le centre et les bords de l'hypercube δ. Cette quantité est appliquée dans toutes les directions $(e_i)_{1 \leqslant i \leqslant N_x}$. La fonction est donc évaluée sur les points $c_1 \pm \frac{1}{3}\delta e_i$, $i = 1, \cdots, N_x$. Dès la première itération, il y a donc $2N_x + 1$ évaluations de f réalisées.



3. Une fois que les divisions ont été faites, il faut ensuite identifier les hyper-rectangles potentiellement optimaux. Pour cela, la condition suivante est proposée par [Finkel et Kelley, 2004]. Soit $\varepsilon > 0$ et soit f_{min} la valeur minimale de la fonction connue à l'itération courante. Un hyper-rectangle j est potentiellement optimal s'il existe un $\tilde{K} > 0$ tel que :

$$\begin{cases} f(\boldsymbol{c}_j) - \tilde{K}\delta_j \leqslant f(\boldsymbol{c}_i) - \tilde{K}\delta_i & \forall i \\ f(\boldsymbol{c}_j) - \tilde{K}\delta_j \leqslant f_{min} - \varepsilon \left| f_{min} \right| & \forall i \end{cases}$$

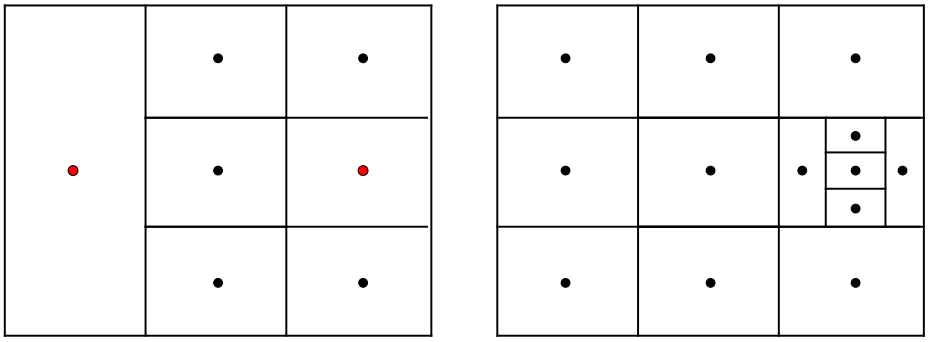


FIGURE 5.3 – Illustration de l'échantillonnage de l'espace par DIRECT. Les hyper-rectangles optimaux sont illustrés avec leur centre en rouge à l'étape 2 à gauche. Ils sont donc divisés selon la règle du point 3 sur la figure de droite.

avec $\boldsymbol{c}_j$ le centre de l'hyper-rectangle j et δ_j est la distance du centre avec les sommets de ce rectangle. Elle est inspirée des algorithmes lipschitziens, ce qui justifie le nom de l'étude de Jones.

4. Une fois que les hyper-rectangles optimaux ont été détectés (points rouges sur le graphe précédent et sur la figure 5.3), ces derniers sont redivisés en hyper-rectangles de taille plus réduite. Pour cela, la fonction objectif f est évaluée deux fois le long des dimensions où l'écart δ_i entre le centre et l'arête de l'hyper-rectangle est la plus longue $c \pm \frac{1}{3}\delta_i e_i$. Dans le cas où l'hyper-rectangle est un hypercube, ce dernier est divisé comme à l'étape 2. Ces deux cas sont illustrés sur la figure 5.3. Si l'hyper-rectangle de gauche est potentiellement optimal, alors il est décomposé uniquement selon la direction où la distance à l'arête est la plus grande, comme illustré sur la figure de droite. Si l'hyper-rectangle de centre droit est également potentiellement optimal, alors il est décomposé comme à la première itération puisque sa distance à l'arête est la même dans toutes les directions.
5. Si le critère d'arrêt est vérifié l'algorithme s'arrête, sinon on retourne au point 3.

Il peut y avoir plus de $2N_x + 1$ évaluations de la fonction à réaliser simultanément à chaque itération de l'algorithme. Cette méthode nécessite donc un nombre important d'appels à la fonction objectif. Dès lors, elle n'est applicable qu'à des fonctions dont le coût d'évaluation est réduit. Toutefois, DIRECT est très adapté au calcul distribué, ce qui peut améliorer le temps de restitution étant donné le nombre important d'appels simultanés à la fonction f qui peut être réalisé. Par ailleurs, le seul paramètre à choisir avec attention est le paramètre K qui sert à déterminer les hyper-rectangles optimaux. Il permet de choisir entre une exploration locale ou globale de l'espace de recherche en focalisant plus ou moins la recherche aux zones où les centres sont à une petite distance de l'arête. Il est donc important de le faire varier au cours de l'algorithme. Cette variation est en général implémentée dans les librairies d'optimisation classique telles que NLOPT [Johnson, 2011].

5.1.2.4 Un exemple d'algorithme d'optimisation global stochastique : évolution différentielle

Les algorithmes évolutionnaires s'inspirent principalement de la biologie : ils consistent à initialiser une population de N_{pop} individus de manière aléatoire, puis à les faire évoluer

selon des règles issues par exemple de la sélection naturelle pour les algorithmes génétiques. Les individus sont des réalisations de jeux de paramètre d'optimisation et ils composent la population : $\mathcal{P} = \left(\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{N_{pop}}\right)$.

Les étapes de fonctionnement des algorithmes évolutionnaires sont décrites sur la figure 5.4. L'idée est de faire évoluer une population d'individus à l'aide d'opérateurs stochastiques inspirés de la biologie comme l'illustre leur nom (mutation, croisement, sélection...). L'étape d'initialisation consiste à générer aléatoirement N_{pop} individus pour construire la population initiale $\mathcal{P}^{k=0}$. Ensuite, chaque itération permet de passer de la population $\mathcal{P}^k$ à une population $\mathcal{P}^{k+1}$. Pour cela, les étapes de mutation et de croisement permettent de générer un vecteur donneur $\boldsymbol{d}$ à partir de la population courante $\mathcal{P}^k$. L'étape de recombinaison permet ensuite d'obtenir une population d'individus candidats $\mathcal{P}^c$. On évalue donc la fonction sur cette population, en général de taille N_{pop}. Enfin, l'étape de sélection permet de construire $\mathcal{P}^{k+1}$ en ne gardant que les individus les plus intéressants parmi la population à l'itération courante $\mathcal{P}^k$ et les individus candidats $\mathcal{P}^c$. Par exemple, CMA-ES est basé sur des générations de populations. Il a été présenté précédemment et rentre dans ce cadre. Une autre technique de ce type régulièrement citée pour ses capacités de résolutions de problèmes très divers est l'algorithme à évolution différentielle. Il a été proposé par Storn et Price dans leur étude [Storn et Price, 1997]. Voici le détail des opérateurs de la figure 5.4 dans le cas de cette méthode :

1. Initialisation : Tirage aléatoire de N_{pop} points pour former la population $\mathcal{P} = \left(\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{N_{pop}}\right)$ puis évaluation de la fonction objectif f sur cette population.
2. Mutation : elle sert à élargir l'espace de recherche. Dans le cas de l'évolution différentielle, tous les membres de la population sont successivement modifiés par un opérateur de mutation. Soit $\boldsymbol{x}^i$ tel que $1 \leqslant i \leqslant N_{pop}$ l'individu que l'on souhaite faire évoluer. Pour cela, trois individus $\boldsymbol{x}^{r1}$, $\boldsymbol{x}^{r2}$ et $\boldsymbol{x}^{r3}$ distincts deux à deux, sont choisis aléatoirement dans la population pour construire un individu donneur $\boldsymbol{d}^i$ de la manière suivante :
$$\boldsymbol{d}^i = \boldsymbol{x}^{r1} + F\left(\boldsymbol{x}^{r2} - \boldsymbol{x}^{r3}\right)$$
Avec $F \in [0, 2]$ le facteur de mutation. Cette étape de construction d'un individu donneur peut varier. Celle-ci, appelée *DE/best/1*, est l'une des plus utilisées puisque simple et n'introduisant qu'un seul paramètre. D'autres variantes sont présentées dans [Mallipeddi et Lee, 2015].
3. Recombinaison : elle permet de combiner les solutions de la population précédente avec le vecteur de don $\boldsymbol{d}^i$. Pour cela, on construit un vecteur $\boldsymbol{u}^i$ candidat au remplacement de $\boldsymbol{x}^i$ en mélangeant aléatoirement les composantes de $\boldsymbol{x}^i$ et du vecteur de don $\boldsymbol{d}^i$. Afin de réaliser ce mélange, on tire N_x valeurs aléatoires $r_1, \cdots, r_{N_x}$ entre $[0, 1]$ et pour chaque composante $1 \leqslant j \leqslant N_x$ on génère le candidat de la manière suivante :
$$u_j^i = \begin{cases} d_j^i \text{ si } r_j \leqslant CR \\ x_j^i \text{ si } r_j > CR \end{cases}$$
Avec $CR \in [0, 1]$ le coefficient de recombinaison à fixer par l'utilisateur. Pour que $\boldsymbol{u}^i \neq \boldsymbol{x}^i$, si aucune des composantes précédentes n'est tombée sur celle du vecteur donneur, il faut aléatoirement échanger l'une des composantes de $\boldsymbol{u}^i$ avec celle de

$\boldsymbol{d}^i$. À cette étape, il faut évaluer la fonction f sur chacun des vecteurs candidats $f(\boldsymbol{u})$.

4. Sélection : Afin de choisir les N_{pop} points à retenir pour la nouvelle population $\mathcal{P}$, on garde celui qui réalise le minimum entre $f(\boldsymbol{u}^i)$ et $f(\boldsymbol{x}^i)$. Ceci signifie que pour tout i tel que $1 \leqslant i \leqslant N_{pop}$, l'individu $\boldsymbol{x}^i$ est choisi ainsi :

$$\boldsymbol{x}^i = \begin{cases} \boldsymbol{u}^i \text{ si } f(\boldsymbol{u}^i) < f(\boldsymbol{x}^i) \\ \boldsymbol{x}^i \text{ sinon} \end{cases}$$

Ensuite, tant que le critère d'arrêt n'est pas vérifié, l'algorithme recommence à l'étape de mutation.

En ce qui concerne le nombre de paramètres à fixer, il est relativement réduit. On peut citer la taille de la population N_{pop} qui doit être au moins égale à 4, le coefficient de recombinaison $CR \in [0, 1]$ et le facteur de mutation $F \in [0, 2]$. Au niveau de l'interprétation de ces coefficients, un CR élevé accélère l'exploration au détriment de l'intensification de la recherche dans une zone. Une valeur faible de ce coefficient a donc pour effet de réduire l'exploration, ce qui augmente le risque de converger vers un minimum local. Le facteur F permet quant à lui d'élargir plus ou moins le domaine de recherche, donc de favoriser l'exploration s'il est élevé ou l'intensification s'il est réduit.

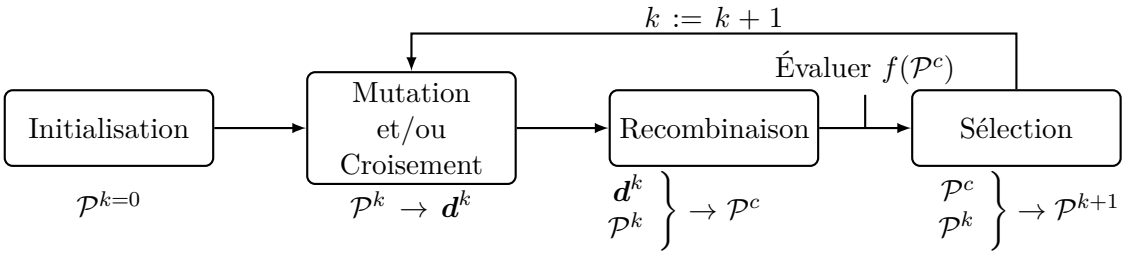


FIGURE 5.4 – Structure des algorithmes évolutionnaires

5.1.3 Algorithmes basés sur les modèles de substitution

En dehors de la méthode de Nelder-Mead, les algorithmes présentés précédemment permettent de trouver un minimum global. En revanche, ils ont pour principal désavantage de nécessiter un nombre d'appels à la fonction important, qui augmente avec la dimension et l'exploration souhaitée. Il peut facilement atteindre le millier d'appels à la fonction à minimiser. Or, les fonctions à optimiser peuvent nécessiter un temps d'exécution conséquent. De ce fait, la possibilité d'utiliser un modèle de substitution $\hat{f}$ à la place du modèle réel f peut avoir pour avantage de diminuer le nombre d'appels à la fonction coûteuse pour atteindre la convergence. L'utilisation d'un modèle de substitution peut par exemple permettre d'estimer le minimum d'une fonction atteint en se donnant un budget fixé d'appels à cette dernière. Pour cela, on peut enrichir séquentiellement le plan d'expériences du métamodèle avec le point le plus informatif jusqu'à atteindre le budget donné. Il est donc nécessaire de définir l'information apportée par un point dans le cas de la recherche du minimum d'une fonction.

Toutefois, un modèle de substitution n'est pas une reproduction parfaite de la fonction qu'il entend substituer. En particulier, si le modèle de substitution est mal utilisé,

rien ne garantit la convergence vers un minimum, ni local ni global, de la fonction de référence f. Pour illustrer cette possibilité, Jones a comparé dans son étude [Jones, 2001] différentes manières d'utiliser un modèle de substitution en optimisation. Il a montré que certaines procédures intuitives ne convergent que vers des minima locaux de la fonction et peuvent même ne pas converger. Par exemple, l'enrichissement du plan d'expériences comme proposé à l'algorithme 5.2 en rajoutant le point minimisant la réponse du modèle de substitution $\hat{f}$ est intuitif mais n'est pas une utilisation adéquate d'un modèle de substitution en optimisation. En effet, dans le cas général, le minimum du modèle de substitution n'est pas un point critique de la fonction réelle, c'est-à-dire un minimum local ou global. De ce fait, il n'apporte pas une information pertinente. L'algorithme itératif rajoutant le minimum du modèle de substitution peut donc très vite être piégé dans une zone inintéressante pour la vraie fonction. Cette partie présente donc les méthodes existantes dans la littérature permettant d'utiliser un modèle de substitution de manière à assurer la convergence vers le minimum global de la fonction de référence.

5.1.3.1 Un algorithme de recherche globale basé sur le krigeage : Efficient Global Optimization (EGO)

Ce qui explique que l'ajout itératif du point minimisant le modèle de substitution au DOE n'est pas une bonne stratégie, c'est que le modèle de substitution commet une erreur d'approximation. Si cette erreur n'est pas maîtrisée ou estimée, il n'y a aucune assurance que le minimum du modèle de substitution soit un point critique de la fonction de référence. Comme cela a été présenté dans la partie 3.2.4, il est possible d'utiliser l'erreur d'estimation fournie par le krigeage afin d'enrichir séquentiellement le DOE. Toutefois, cet enrichissement n'a pas pour but ici d'améliorer la connaissance de la fonction f sur tout le domaine de définition. Comme à la partie 4.2.3, il est nécessaire de cibler les zones d'intérêt en construisant la fonction d'information $\mathcal{I}\text{nfo}$ apportée par un point et adaptée au problème d'optimisation. Ce point de vue est détaillé dans l'algorithme 5.2 qui est une application de l'algorithme 3.5 au cas de l'optimisation.

Une première manière de définir cette mesure d'information est d'utiliser la borne de confiance inférieure aussi appelée GP-LCB pour *Gaussian Process Lower Confidence Bound algorithm*. Cette méthode est par exemple citée dans le cas d'un problème de maximisation d'une fonction bruitée dans [Srinivas *et al.*, 2010] ou [Snoek *et al.*, 2012]. Cela revient à définir la fonction $\mathcal{I}\text{nfo}$ de la manière suivante :

$$\mathcal{I}\text{nfo}\,(\boldsymbol{x}) = -\left[\hat{f}(\boldsymbol{x}) - k\hat{\sigma}(\boldsymbol{x})\right] \tag{5.2}$$

où $k > 0$ est un paramètre à fixer. Il est souvent pris égal à 1.96 pour que l'équation (5.2) corresponde à la borne de confiance inférieure à 95% de la prédiction par krigeage en tout point $\boldsymbol{x}$. Cependant, le choix de k est très sensible sachant que la vitesse de convergence, et donc le résultat final à budget fixé, en dépend fortement. Il est la traduction du compromis entre exploration et intensification détaillé précédemment.

Afin de ne plus avoir de paramètres de ce type à gérer, Jones [Jones *et al.*, 1998] utilise un critère introduit par Močkus [Močkus, 1975] : l'amélioration espérée (*Expected Improvement* EI en anglais). Pour ce faire, il traduit mathématiquement ce que recherche l'utilisateur lorsqu'il optimise une fonction à partir d'un modèle de substitution : maximiser l'espérance d'améliorer la connaissance du minimum. En utilisant le processus gaussien défini par le krigeage $Y(\boldsymbol{x})$ à l'équation (3.16), ceci se traduit mathématiquement par

l'équation (5.3).

$$\mathcal{I}\text{nfo}\,(\boldsymbol{x}) = \mathbb{E}\left[\max\left(0, Y(\boldsymbol{x}) - \min_{\boldsymbol{x}\in DOE} f(\boldsymbol{x})\right)\middle| DOE\right] \tag{5.3}$$

Or, dans le cas du krigeage, cette espérance peut se calculer analytiquement puisque l'unique source d'aléa, $Y(\boldsymbol{x})$, suit une loi normale $\mathcal{N}\left(\hat{f}(\boldsymbol{x}), \hat{\sigma}(\boldsymbol{x})\right)$ conditionnellement au plan d'expériences. En notant $f_{min} = \min_{\boldsymbol{x}\in DOE} f(\boldsymbol{x})$ le minimum sur le plan d'expériences courant, le critère de l'équation (5.3) se réécrit donc :

$$\begin{aligned}\mathbb{E}\left[\max\left(0, Y(\boldsymbol{x}) - f_{min}\right)\middle| DOE\right] &= \int_{f_{min}}^{\infty} (y - f_{min})\, \varphi_{\hat{f}(\boldsymbol{x}),\hat{\sigma}(\boldsymbol{x})}(y)\ \ dy \\ &= \int_{f_{min}}^{\infty} y\varphi_{\hat{f}(\boldsymbol{x}),\hat{\sigma}(\boldsymbol{x})}(y)\ \ dy - f_{min}\int_{f_{min}}^{\infty} \varphi_{\hat{f}(\boldsymbol{x}),\hat{\sigma}(\boldsymbol{x})}(y)\ \ dy\end{aligned}$$

En intégrant par partie l'intégrale de gauche (le résultat du calcul de cette intégrale est donné à l'équation (B.1) de l'annexe B) et en reconnaissant le membre de droite comme l'intégrale de la densité d'une loi normale, que l'on peut donc exprimer par sa fonction de répartition, on obtient donc pour le critère EI :

$$\mathcal{I}\text{nfo}\,(\boldsymbol{x}) = \left[f_{min} - \hat{f}\,(\boldsymbol{x})\right]\Phi\left[\frac{f_{min} - \hat{f}\,(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right] + \hat{\sigma}(\boldsymbol{x})\varphi_{(0,1)}\left[\frac{f_{min} - \hat{f}\,(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right] \tag{5.4}$$

Remarque sur la différence entre ces deux critères : La figure 5.5 permet d'illustrer la différence entre les critères LCB et EI. Alors que le critère d'amélioration espérée permet de rajouter le point dont la densité de la sortie prédite par krigeage donne le plus de poids à la zone en dessous du minimum courant, ce qui est représenté par la zone verte, le critère LCB ne se base quant à lui que sur la moyenne et la variance de la prédiction en chaque point et ne prend pas en compte la valeur du minimum courant.

Dès lors, l'EI permet d'utiliser l'intégralité des informations que fournit un modèle de krigeage en chaque point $\boldsymbol{x}$ alors que LCB ne se limite qu'à deux paramètres de forme de la densité, sans prendre en compte la probabilité d'être en dessous de ce minimum courant. Or, ce qui est réellement important afin de choisir le meilleur point à rajouter au DOE est l'intégrale de la part de densité située en dessous de ce minimum (représentée par la partie remplie en vert sur le graphique). Par exemple, sur cette figure, le point qui serait rajouté par le critère LCB serait le point $x = 0$ alors que sa probabilité d'être en dessous du minimum est réduite comparée au point $x \simeq 5$.

En revanche, l'une des raisons de l'utilisation du critère LCB est sa capacité à mieux gérer les fonctions f bruitées, comme détaillé dans l'étude [Srinivas *et al.*, 2010].

5.1.3.2 Autres algorithmes d'optimisation par modèles de substitution

Les méthodes basées sur l'enrichissement séquentiel d'un modèle de krigeage sont les plus répandues dans la littérature. Toutefois, il existe d'autres méthodes entièrement basées sur les modèles de substitution qui s'appellent algorithmes à région de confiance (aussi appelées *Trust Region* en anglais). La différence principale avec les algorithmes basés sur l'enrichissement séquentiel du DOE est l'utilisation de modèles de substitution locaux et non plus globaux. Ceci signifie que le modèle de substitution n'est plus utilisé sur le domaine de définition entier $\mathcal{D}_x$ mais uniquement sur une zone réduite de ce domaine. Plus

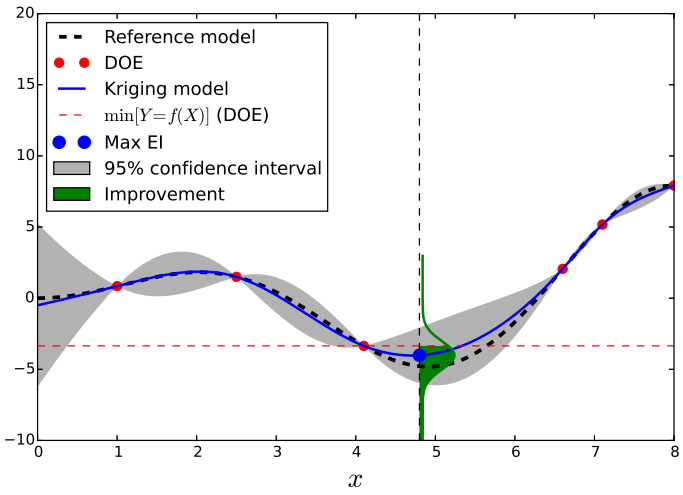


FIGURE 5.5 – Illustration de l'amélioration et des bornes de confiance inférieures à 95%

Algorithme 5.2 Optimisation globale par enrichissement du DOE d'un modèle de krigeage

Input : La taille du DOE initial N^0_{DOE}, le budget total pour le DOE N_{DOE}, le critère d'enrichissement $\mathcal{I}$nfo

Output : Le minimum estimé de la fonction f_{min} et le point où il se réalise $\boldsymbol{x}^{min}$

· Générer les N^0_{DOE} entrées du DOE initial $\boldsymbol{X}^0$ par LHS;
· Évaluer $\boldsymbol{Y}^0 = f(\boldsymbol{X}^0) \rightarrow DOE^0$;
· Soit $N_{courant} := N^0_{DOE}$ et $DOE^{courant} := DOE^0$;
while $N_{courant} < N_{DOE}$ **do**
  · Construire $\hat{f}$ avec $DOE^{courant}$;
  · Choisir $\boldsymbol{x}^\star = \text{argmax}_{\boldsymbol{x}\in\mathcal{D}_x}\ \mathcal{I}\text{nfo}(\boldsymbol{x})$;
  · Évaluer $f(\boldsymbol{x}^\star)$;
  · $DOE^{courant} = \{DOE^{courant}, (\boldsymbol{x}^\star, f(\boldsymbol{x}^\star))\}$;
end
· $f_{min} = \min_{\boldsymbol{x}\in DOE} f(\boldsymbol{x})$;
· $\boldsymbol{x}^{min} = \underset{\boldsymbol{x}\in DOE}{\text{argmin}} f(\boldsymbol{x})$;

précisément, le modèle de substitution est utilisé afin de trouver le minimum local de cette zone réduite. En fonction de la position de ce minimum local, la région de confiance peut se déplacer, s'agrandir ou bien se réduire, de la même manière que dans le cas des algorithmes locaux comme CMA-ES présentés en section 5.1.2.1. Le plus fréquemment, cette méthode utilise des régressions polynomiales mais d'autres approches basées sur les RBF ou le krigeage existent. Par exemple, un exemple récent proposé par Regis [Regis, 2016] mélange région de confiance et critère d'amélioration espérée du krigeage. Un autre exemple basé quant à lui sur les RBF peut être trouvé dans l'étude [Wild et Shoemaker, 2013]. Ces techniques peuvent se trouver sous le nom de ORBIT *Optimization by Radial Basis functions In Trust Regions*. L'étude de Wild fournit d'ailleurs une preuve de convergence de ces méthodes.

Par ailleurs, il existe d'autres façons d'utiliser un modèle de substitution en opti-

misation. Par exemple, beaucoup d'études s'intéressent au mélange de techniques évolutionnaires avec l'utilisation de modèle de substitution. Notamment, Loshchilov dans [Loshchilov *et al.*, 2013] combine CMA-ES avec un modèle de substitution en réalisant certaines générations sur le modèle de substitution et d'autres sur la vraie fonction. Les générations réalisées sur la fonction de référence f permettent de contrôler le nombre de générations réalisées sur le modèle de substitution en utilisant l'erreur que commet le métamodèle sur les points calculés sur f. Autre technique combinant CMA-ES et modèle de substitution, Mohammadi dans son étude [Mohammadi *et al.*, 2015] initialise une recherche locale par CMA-ES à l'aide de EGO présenté à l'algorithme 5.2 afin de combiner les capacités d'exploration de ce dernier avec l'intensification que permet CMA-ES. Enfin, il existe d'autres études combinant méthodes évolutionnaires et modèle de substitution, par exemple [Mallipeddi et Lee, 2015] le fait avec l'algorithme à évolution différentielle. Dans cette étude, le modèle de substitution est utilisé afin de tester différentes combinaisons de paramètres F et CR tout en affinant sa précision avec les générations.

5.2 Optimisation Multi-objectif

Dans cette partie, la résolution de problèmes d'optimisation caractérisés par la présence de plusieurs critères à minimiser $f_1, \cdots, f_{N_{obj}}$ ($N_{obj} \geqslant 2$) est développée. Soit $\boldsymbol{F}(\boldsymbol{x}) = \left(f_1(\boldsymbol{x}), \cdots, f_{N_{obj}}(\boldsymbol{x})\right)$ le vecteur des objectifs au point $\boldsymbol{x}$. Alors qu'avec un unique objectif, il est facile de comparer deux solutions en les ordonnant selon la valeur prise par la fonction coût en ces points, il n'est pas toujours évident de comparer deux solutions dans le cas multi-objectif. Par exemple avec deux objectifs, comment comparer deux solutions $\boldsymbol{x}$ et $\boldsymbol{x}'$ telles que $f_1(\boldsymbol{x}) < f_1(\boldsymbol{x}')$ et $f_2(\boldsymbol{x}) > f_2(\boldsymbol{x}')$? Afin de pouvoir définir ce qu'est un optimum malgré l'impossibilité dans le cas général de pouvoir comparer toutes les solutions entre elles, il est nécessaire d'introduire la dominance de Pareto. C'est une relation d'ordre partielle qui permet de généraliser au cas multi-objectif la relation d'ordre totale qu'est le « plus petit que » ($<$) avec un seul objectif.

Définition de la dominance de Pareto : Soit $(\boldsymbol{x}, \boldsymbol{x}') \in \mathcal{D}_x^2$. On dit que $F(\boldsymbol{x})$ domine $F(\boldsymbol{x\prime})$, ce qui s'écrit $\boldsymbol{F}(\boldsymbol{x}) \prec \boldsymbol{F}(\boldsymbol{x}')$ dans le cas de la recherche du minimum, si et seulement si

$$\begin{cases} \forall i \in \{1, \cdots, N_{obj}\}, & f_i(\boldsymbol{x}) \leqslant f_i(\boldsymbol{x}') \\ \exists j \in \{1, \cdots, N_{obj}\}, & f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}') \end{cases}$$

Cette dominance est illustrée en figure 5.6 dans le cas de deux objectifs.

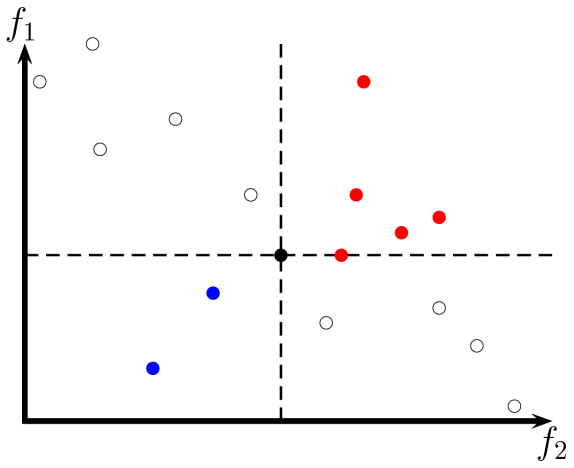


FIGURE 5.6 – Illustration de la dominance de Pareto dans le cas de deux objectifs. Le point noir domine les points rouges, est dominé par les points bleus et n'est pas comparable au sens de Pareto avec les points blancs.

Définition de l'optimalité de Pareto : un point $\boldsymbol{x} \in \mathcal{D}_x$ est Pareto-optimal s'il n'existe pas d'autres points $\boldsymbol{x}' \in \mathcal{D}_x$ tel que $\boldsymbol{F}(\boldsymbol{x}) \succ \boldsymbol{F}(\boldsymbol{x}')$. En d'autres termes, un point Pareto-optimal est un point dont le vecteur des objectifs $F(\boldsymbol{x})$ est non-dominé.

Définition du front de Pareto optimal : Sachant que la dominance de Pareto est une relation d'ordre partielle, certains points ne sont pas comparables par la dominance de Pareto comme illustré par les points blancs sur la figure 5.6. De

ce fait, il peut exister plusieurs points Pareto-optimaux. On appelle donc l'ensemble des points Pareto-optimaux le front de Pareto optimal (noté $PF^\star$), et défini par $\boldsymbol{X}^{PF^\star} = \{\boldsymbol{x} \in \mathcal{D}_x \ : \nexists \boldsymbol{x}' \in \mathcal{D}_x \text{ tel que } \boldsymbol{F}(\boldsymbol{x}) \succ \boldsymbol{F}(\boldsymbol{x}')\}$ et les sorties en ces points : $\boldsymbol{Y}^{PF^\star} = \boldsymbol{F}\left(\boldsymbol{x}^{PF^\star}\right) = \left\{f_1(\boldsymbol{x}^{PF^\star}), \cdots, f_{N_{obj}}(\boldsymbol{x}^{PF^\star})\right\}$. Ce sont donc l'ensemble des points non-dominés et les solutions recherchées du problème d'optimisation.

Prise en compte de plusieurs critères en présence d'un *a priori*. La dominance de Pareto permet donc de définir ce que signifie un optimum en présence de plusieurs objectifs : un ensemble de compromis optimaux et non comparables entre eux sans hiérarchisation des objectifs. Toutefois, il arrive que l'utilisateur ne souhaite pas connaître cet ensemble dans son intégralité, par exemple s'il ne recherche qu'une seule solution. Dans ce cas-là, s'il est de plus capable de hiérarchiser les objectifs à l'aide d'*a priori* disponibles sur ceux-ci, il est possible de reformuler le problème en un problème mono-objectif avec ou sans contraintes. Voici une liste non exhaustive d'alternatives à l'optimisation simultanée de plusieurs critères provenant des études [Zhang et Li, 2007] ou de la revue des méthodes de résolution d'un problème avec plusieurs critères fournie par [Marler et Arora, 2004] :

— **Introduire une fonction utilité mono-objectif** où tous les objectifs sont sommés puis pondérés par des réels $a_i > 0$ tels que $\sum_{i=1}^{N_x} a_i = 1$. Ces pondérations servent à refléter les choix provenant des *a priori* disponibles. Cette technique s'appelle *Weighted Sum* dans la littérature. Par exemple, ceci revient à résoudre :

$$\min_{x \in \mathcal{D}_x} f(\boldsymbol{x}) \text{ avec } f(\boldsymbol{x}) = \sum_{i=1}^{N_x} a_i f_i(\boldsymbol{x}) \tag{5.5}$$

les objectifs f_i pouvant être rendus sans dimension à l'aide d'une valeur caractéristique afin d'être comparables même dans le cas où ils traduisent des quantités de nature différente, comme une longueur et une masse.
Ces formulations ont toutefois un certain nombre de limites : il peut être difficile de choisir les coefficients a_i, le problème d'optimisation ainsi défini n'a pas toujours une solution acceptable et, dans le cas général, cette formulation ne permet pas de trouver toutes les zones du front de Pareto dans l'espace des objectifs. Toutefois, pour compenser le premier défaut il est possible de faire évoluer les poids $\boldsymbol{a}$ au cours de l'optimisation.

— **Introduire une fonction utilité mono-objectif** plus complexe telle que la pondération de Tchebycheff aussi appelée *Weighted Min-Max*. Elle consiste à minimiser le maximum de tous les objectifs pondérés, c'est-à-dire :

$$\min_{x \in \mathcal{D}_x} f(\boldsymbol{x}) \text{ avec } f(\boldsymbol{x}) = \max_{1 \leqslant i \leqslant N_{obj}} \{a_i \, |f_i(\boldsymbol{x}) - f_i^\star|\} \tag{5.6}$$

avec $\left(f_1^\star, \cdots, f_{N_{obj}}^\star\right) \in \mathbb{R}^{N_{obj}}$ est le vecteur des solutions des problèmes mono-objectifs c'est-à-dire $\forall i \in \{1, \cdots, N_{obj}\}$, $f_i^\star = \min_{x \in \mathcal{D}_x} f_i(\boldsymbol{x})$.
Cette formulation présente l'avantage d'être une condition nécessaire et suffisante pour que la solution soit Pareto-optimale. Néanmoins, le choix des poids a_i n'est pas toujours évident, notamment si le front entier est recherché. Comme précédemment, il est possible de faire évoluer ces poids au cours de l'optimisation.

— **Utiliser une méthode lexicographique** consistant à ranger les objectifs par ordre d'importance et en résolvant de manière itérative des problèmes mono-objectifs

sous contraintes du type :

$$\min_{x \in \mathcal{D}_x} f_i(\boldsymbol{x}) \text{ s.c } f_j(\boldsymbol{x}) \leqslant f_j(\boldsymbol{x}^\star) \quad j \neq i \tag{5.7}$$

en commençant par l'objectif le plus important : $f_i(\boldsymbol{x}^\star) = \min_{x \in \mathcal{D}_x} f_i(\boldsymbol{x})$. Cette formulation présente l'avantage d'être une condition nécessaire et suffisante pour que la solution soit Pareto-optimale. Cependant, les contraintes sont très restrictives et il est donc compliqué de trouver une solution admissible pour chacun des problèmes. De plus, cette méthode étant itérative, elle est coûteuse et difficilement parallélisable.

— En simplifiant le point précédent, il est également possible de se ramener à un problème mono-objectif avec contraintes en ne **minimisant que l'objectif le plus important et en mettant les autres objectifs en contraintes** :

$$\min_{x \in \mathcal{D}_x} f_i(\boldsymbol{x}) \text{ s.c } f_j(\boldsymbol{x}) \leqslant \bar{f}_j \quad j \neq i \tag{5.8}$$

avec $\bar{f}_j$ étant un *a priori* décrivant la valeur à ne pas dépasser par les autres objectifs moins importants. De la même manière que pour la méthode lexicographique, cela peut conduire à un espace de solution admissible très restreint voire vide ce qui rend la résolution potentiellement difficile.

D'une manière générale, ces méthodes ne conduisent pas à la même solution et chacune permet de prendre en compte des *a priori* de nature différente. Dans le cas où l'utilisateur n'a aucun *a priori* disponible ou s'il souhaite connaître l'ensemble des solutions optimales, aucune de ces méthodes n'est utilisable en l'état et il est nécessaire d'utiliser des algorithmes multi-objectifs globaux, c'est-à-dire cherchant le front de Pareto entier. Ceci permet au décideur de faire son choix parmi un ensemble de compromis optimaux non comparables. Ces méthodes sont détaillées dans la première partie, notamment celles basées sur les algorithmes génétiques du type NSGA-II. Enfin, l'utilisation de modèles de substitution pour la résolution de problèmes multi-objectifs est également détaillée. Comme en mono-objectif, cette méthode permet de réduire le nombre d'appels nécessaires aux objectifs.

5.2.1 Algorithmes multi-objectif globaux

Le front de Pareto optimal ($PF^\star$) fournit aux ingénieurs un ensemble de solutions optimales non comparables, c'est-à-dire améliorant au moins l'un des objectifs par rapport aux autres points Pareto-optimaux. Cela permet donc à l'ingénieur de choisir le compromis qui correspond le plus à son besoin.

Afin de calculer le front de Pareto entier, il est nécessaire d'explorer intensivement l'espace des objectifs : alors qu'en mono-objectif, les zones à explorer sont celles où la valeur de l'objectif est minimum, en multi-objectifs, un point quelconque pour l'un des objectifs peut tout à fait être le point critique des autres critères. De la même manière, des points non critiques ne représentant le minimum d'aucun des objectifs peuvent tout à fait appartenir à $PF^\star$.

Cette exploration peut se faire de deux manières : soit en utilisant des algorithmes basés sur la résolution de plusieurs problèmes mono-objectifs et en faisant varier les poids permettant cette scalarisation. C'est par exemple le cas de la méthode NBI pour *Normal*

Boundary Intersection [Das et Dennis, 1998] développée par Das et Dennis. Cette technique a l'avantage d'être applicable quel que soit le nombre d'objectifs. Toutefois, elle nécessite de résoudre des problèmes sous contraintes. Le nombre de résolutions de ce type peut d'ailleurs être important puisqu'à chaque point souhaité dans le front, il faut réaliser une nouvelle résolution. De plus, il faut rajouter la recherche des optima mono-objectifs selon chaque critère avant de chercher les autres points du front, ce qui nécessite N_{obj} optimisations mono-objectifs. Enfin, les discontinuités du front sont mal prises en compte par ce type de méthodes.

Les autres méthodes consistent à mettre en place des algorithmes évolutionnaires dont la sélection est basée sur la dominance de Pareto.

Méthodes évolutionnaires - L'exemple de NSGA-II

Afin d'avoir un front discrétisé plus finement et de mieux explorer les différentes objectifs, les méthodes évolutionnaires présentées dans le cas mono-objectif, par exemple par la figure 5.4, sont une alternative intéressante en multi-objectif. En effet, elles permettent de prendre en compte les discontinuités du front ou des fonctions objectif, elles permettent aussi de pouvoir travailler sur des domaines faisables disjoints ou encore elles peuvent être utilisées avec des fonctions bruitées.

Par ailleurs, leur fonctionnement est intrinsèquement adapté à la prise en compte de différents objectifs : premièrement parce qu'elles explorent largement l'espace des entrées et donc les fonctions objectif. Deuxièmement parce que la notion d'évolution biologique, notamment génétique, permet de favoriser la diversité parmi une population, ce qui est justement le but dans le cas de l'optimisation multi-objectif où il ne faut pas favoriser une zone du front par rapport à une autre. Cette propriété s'appelle le nichage et peut être prise en compte en choisissant les bons opérateurs de sélection.

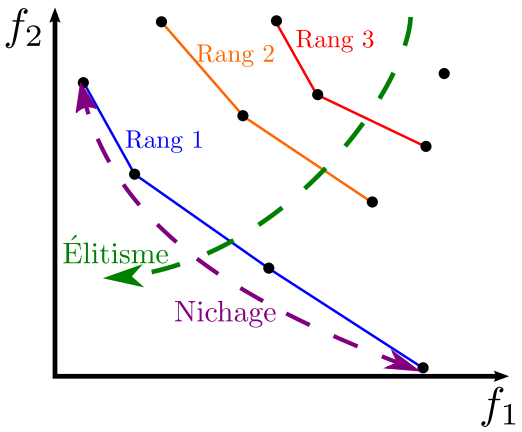


FIGURE 5.7 – Illustration des critères de sélection des méthodes évolutionnaires

Il existe de nombreuses techniques évolutionnaires pour l'optimisation multi-objectif. Certaines sont basées sur l'adaptation au cas multi-objectif d'algorithmes mono-objectifs. Par exemple, les essaims particulaires appelés MOPSO [Coello Coello et Lechuga, 2002] ou CMA-ES avec MO-CMA-ES [Igel *et al.*, 2007] ont été développés en modifiant l'opérateur de sélection afin de l'adapter à la présence de plusieurs critères, par exemple en utilisant la notion d'hypervolume, qui est définie en section 5.2.2, dans le cas de MO-CMA-ES.

D'autres algorithmes sont basés sur la transformation du problème multi-objectif en la résolution de plusieurs problèmes mono-objectifs, par exemple à l'aide de la pondération de Tchebycheff détaillée à l'équation (5.6). Cette méthode s'appelle MOEA/D pour *Multiobjective Evolutionary Algorithm Based on Decomposition* [Zhang et Li, 2007] et elle consiste à résoudre simultanément N_{pop} sous-problèmes formulés à l'aide de N_{pop} valeurs différentes des coefficients $\boldsymbol{a}$ pour chaque sous-problème. Ces N_{pop} sous-problèmes sont ensuite résolus simultanément en une seule exécution de l'algorithme évolutionnaire. Afin de ne pas perdre de solutions en cours de résolution, les meilleures solutions à l'itération courante sont stockées dans une population à part.

Enfin, les méthodes les plus populaires sont quant à elles basées sur un mélange d'élitisme et de nichage dans leur étape de sélection. L'élitisme consiste à ne garder que les solutions qui sont le moins dominées, par exemple en classant les solutions par rang. Le nichage consiste à garder de la diversité dans les solutions en favorisant celles qui sont le plus éloignées d'autres solutions dans l'espace des objectifs. Ces deux critères sont représentés sur la figure 5.7. On peut par exemple citer SPEA-II [Zitzler *et al.*, 2001] ou NSGA-II [Deb *et al.*, 2002] qui sont deux algorithmes populaires car fournissant le meilleur résultat dans une grande diversité de cas tests selon l'étude de Deb [Deb, 2011]. Alors que SPEA-II est basé sur le stockage des meilleures solutions, NSGA-II travaille à population constante.

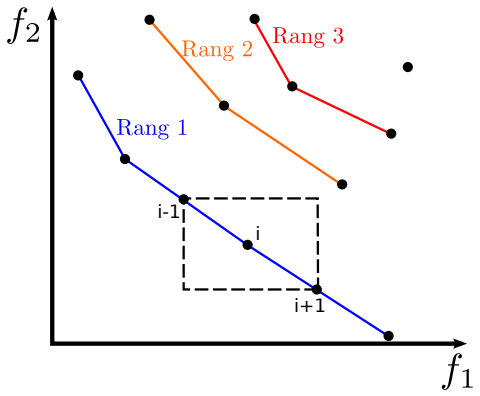
NSGA-II (*Non-dominated Sorting Genetic Algorithm*) s'inspire de la génétique en travaillant sur le génotype (c'est-à-dire les coordonnées des entrées) des meilleurs individus à l'itération précédente pour générer à chaque itération les nouveaux individus. Afin de caractériser ce qu'est une bonne solution et détecter les meilleures, l'opérateur de sélection se base sur le phénotype, c'est-à-dire la valeur des objectifs pour ce jeu de coordonnées. Elles permettent de garder les solutions favorisant l'élitisme et le nichage. Les étapes de cet algorithme sont représentées sur la figure 5.10. En voici le détail :

— **L'initialisation** permet de générer aléatoirement sur tout le domaine $\mathcal{D}_x$ une population de N_{pop} individus $\mathcal{P} = \left(\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{N_{pop}}\right)$. Ensuite, comme pour l'algorithme à évolution différentielle (cf figure 5.4), NSGA-II fait évoluer la population grâce à des opérateurs stochastiques de croisement et de mutation. Il sont décomposés de la manière suivante :

1. **Choix des parents** parmi la population initiale de $\frac{N_{pop}}{2}$ individus par tournoi binaire. Cela consiste à tirer aléatoirement deux individus différents de la population $\mathcal{P}$ $\boldsymbol{x}^{i1}$ et $\boldsymbol{x}^{i2}$ et à garder le meilleur des deux. En multi-objectif, comme cela est représenté sur la figure 5.7, la qualité d'une solution se mesure d'abord par élitisme en regardant le rang des deux individus, c'est-à-dire le numéro du front auquel chacune des deux solutions appartient. Le rang est calculé itérativement : les points non dominés sont les points de rang 1. Ils sont ensuite retirés et les points de rang 2 sont de nouveau les points non dominés parmi les points restants, et ainsi de suite jusqu'à ce que tous les points soient classés. Si les deux points à comparer sont de même rang, des techniques de nichage sont développées afin de favoriser la diversité. Pour cela, NSGA-II utilise la *crowding distance* ou distance de nichage. Elle permet de calculer la proximité d'un point à ses voisins du front du même rang que lui.

En ordonnant les valeurs des entrées selon chaque objectif, on obtient pour la distance de nichage en un point i d'un rang quelconque r :

$$d_r(i) = \begin{cases} \infty \text{ Si } \boldsymbol{x}^i \text{ au bord} \\ \sum_{k=1}^{N_{obj}} \frac{f_k(\boldsymbol{x}^{(i+1)}) - f_k(\boldsymbol{x}^{(i-1)})}{f_k^{max} - f_k^{min}} \end{cases} \tag{5.9}$$



Le fait de conserver les points ayant la distance de nichage la plus grande à rang fixé permet de favoriser la diversité dans la population et ainsi de tendre vers l'exploration du front entier. Afin d'obtenir $\frac{N_{pop}}{2}$ parents, cette étape est répétée $\frac{N_{pop}}{2}$ fois.

2. À partir des parents générés précédemment, il faut à présent **générer les descendants** (*offspring* en anglais). Ceci peut se faire soit par mutation (un parent donne un descendant) soit par croisement (deux parents donnent deux descendants). Le choix entre les deux est réalisé de manière aléatoire. La probabilité d'appliquer le croisement p_c est donnée par l'utilisateur. La probabilité de mutation est donc $1 - p_c$.
   — La mutation consiste à générer un individu enfant $\boldsymbol{x}^c$ en modifiant les coordonnées d'un individu parent $\boldsymbol{x}^p$. Le type de mutation classiquement utilisé dans NSGA-II est la mutation polynomiale [Raghuwanshi et Kakde, 2004] et consiste à générer le descendant de manière aléatoire. Pour cela, on utilise une variable aléatoire δ_i générée pour chaque composante j, $1 \leqslant j \leqslant N_x$:

$$x_j^c = x_j^p + \delta_j \Delta_{max\,j} \tag{5.10}$$

   Ici, $\Delta_{max\,j}$ est l'écart à la borne de la composante x_j^p. La densité de la variable δ_j est déterminée à l'aide d'un paramètre d'entrée fourni par l'utilisateur c_{mut}. La représentation graphique de la dépendance de la densité à ce paramètre est donnée en figure 5.8. Grâce à c_{mut}, l'utilisateur peut donc choisir d'explorer de manière plus ou moins importante l'espace des entrées. Par exemple si c_{mut} est grand, le descendant est généré à proximité du parent et l'opérateur de mutation sert à l'intensification. À l'inverse, s'il est proche de 0, la mutation permet de favoriser l'exploration.
   — Le croisement consiste à mélanger de manière stochastique le génotype de deux parents $\boldsymbol{x}^{p1}$ et $\boldsymbol{x}^{p2}$ de sorte à obtenir deux descendants $\boldsymbol{x}^{c1}$ et $\boldsymbol{x}^{c2}$. La technique la plus utilisée s'appelle SBX pour *Simulated Binary Crossover* [Raghuwanshi et Kakde, 2004]. Elle consiste à générer aléatoirement une variable aléatoire β_j pour chaque composante et à utiliser cette variable pour calculer les composantes des descendants à partir de celles des parents. La densité de β_j est déterminée par l'utilisateur à l'aide du coefficient c_{cross} à fixer. Cette densité est représentée selon différentes valeurs de c_{cross} en figure 5.9. Une fois le paramètre β_j généré, les descendants sont calculés à

Calcul de la densité h_δ de δ en fonction de c_{mut} (donné dans [Raghuwanshi et Kakde, 2004]) :

$$h_\delta(\delta) = 0.5\,(c_{mut}+1)\,(1-|\delta|)^{c_{mut}} \quad (5.11)$$

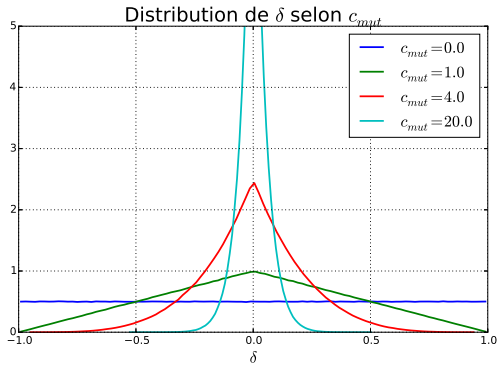


FIGURE 5.8 – Illustration de la densité de δ selon la valeur du paramètre c_{mut} fixé par l'utilisateur

Calcul de la densité h_β de β en fonction de c_{cross} (donné dans [Raghuwanshi et Kakde, 2004]) :

$$h_\beta(\beta) = \begin{cases} 0.5\,(c_{cross}+1)\,\beta^{c_{cross}} & \text{si } \beta \leqslant 1 \\ 0.5\,(c_{cross}+1)\,\frac{1}{\beta^{c_{cross}+2}} & \text{si } \beta > 1 \end{cases} \quad (5.13)$$

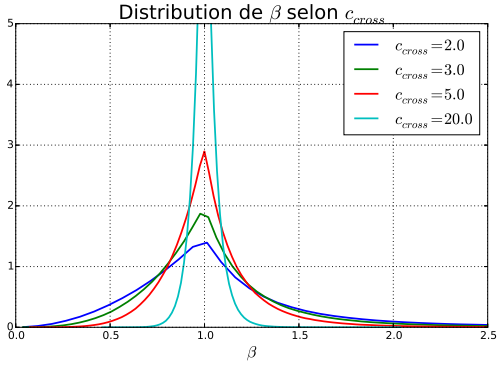


FIGURE 5.9 – Illustration de la densité de β selon la valeur du paramètre c_{cross} fixé par l'utilisateur

l'aide de l'équation (5.12).

$$\begin{cases} x_j^{c1} = \dfrac{x_j^{p1}+x_j^{p2}}{2} + \beta_j \dfrac{x_j^{p1}-x_j^{p2}}{2} \\ x_j^{c1} = \dfrac{x_j^{p1}+x_j^{p2}}{2} - \beta_j \dfrac{x_j^{p1}-x_j^{p2}}{2} \end{cases} \quad (5.12)$$

Si $\beta < 1$ les descendants se trouvent entre les deux parents alors que si $\beta > 1$ ils se trouvent entre l'un des deux parents et les bornes du domaine. Enfin, plus β est proche de 1 et plus les descendants sont à proximité des deux parents. Comme dans le cas de la mutation, il est donc possible pour l'utilisateur de favoriser l'exploration ou l'intensification selon la valeur donnée au coefficient c_{cross}. Plus la valeur de c_{cross} est élevée et plus l'intensification est favorisée. Inversement, plus c_{cross} est proche de 0, plus l'exploration est favorisée.

Pour résumer l'étape de génération de nouveaux individus, il y a trois paramètres à choisir : la part de croisement ou de mutation p_c et les deux paramètres

de mutation et de croisement c_{mut} et c_{cross} qui servent à définir un compromis entre l'exploration ou l'intensification. Pour le choix de p_c, il est d'usage de favoriser le croisement au détriment de la mutation puisque cela favorise la diversité de la population. En ce qui concerne les deux paramètres de mutation et de croisement, ils ont pour but de réaliser un compromis entre exploration et intensification, comme le montrent les figures 5.9 et 5.8. Il est par exemple d'usage d'utiliser l'opérateur de mutation pour l'intensification et le croisement pour l'exploration.

— Enfin, afin de ne garder que les individus améliorant la population et de garantir la non-régression, **la sélection et la recombinaison** se font en mélangeant la population de descendants avec la population initiale, ce qui fait une population intermédiaire de $\sim 2N_{pop}$ individus. Ensuite, à l'aide du rang et de la distance de nichage seulement N_{pop} individus sont conservés afin d'obtenir la population finale.

— Remarque sur la convergence : cet algorithme présente l'avantage de combiner une exploration intensive, à l'aide des opérateurs de croisement et de mutation, à la garantie de non-régression que permet l'opérateur de sélection basé sur la distance de nichage et le rang. Cela implique que la convergence est assurée. Toutefois, le nombre de générations nécessaires à cette convergence est une inconnue qui peut être théoriquement très importante.

L'une des limites de la recherche de fronts de Pareto optimaux est qu'elle est mal adaptée au cas où le nombre de fonctions objectif à minimiser est supérieur à trois. Au-delà de cette limite, il est difficile d'interpréter un front puisqu'il n'est pas représentable. En outre, plus le nombre d'objectifs est grand, plus la probabilité qu'un point appartienne au front de Pareto est grande, ce qui affaiblit la méthode de sélection basée sur la dominance des algorithmes présentés (voir [Ishibuchi *et al.*, 2008]). Il est donc nécessaire de redéfinir ce qu'est un optimum en présence d'un nombre d'objectifs trop important. C'est notamment ce qui est proposé dans le développement de NSGA-III [Deb et Jain, 2014].

5.2.2 Algorithmes basés sur les modèles de substitution

De la même manière que pour les algorithmes mono-objectifs, l'utilisation de modèles de substitution permet de réduire le nombre d'appels aux fonctions objectif. Cette utilisation peut de nouveau se faire de deux manières : soit en utilisant un modèle de substitution conjugué à un algorithme multi-objectif existant, soit en utilisant une technique d'enrichissement séquentiel uniquement basée sur le choix du nouveau point à rajouter au plan d'expériences. Cet enrichissement séquentiel donne notamment la possibilité de remplacer la résolution du problème multi-objectif sur des fonctions coûteuses en un problème d'optimisation mono-objectif sur une fonction peu coûteuse, telle que l'amélioration espérée estimée par krigeage.

5.2.2.1 Modèles de substitution et algorithmes évolutionnaires

Comme dans le cas mono-objectif, il est possible d'utiliser un modèle de substitution en complément d'un algorithme évolutionnaire. Ceci signifie que le modèle de substitution n'est pas utilisé pour diriger l'optimisation mais qu'il permet de diminuer le nombre d'appels aux fonctions objectif en les remplaçant. Ceci implique que le résultat qui est retourné par ce type de méthodes est la population obtenue sur les prédictions des modèles de substitution lors de la dernière génération de l'algorithme évolutionnaire. De ce fait,

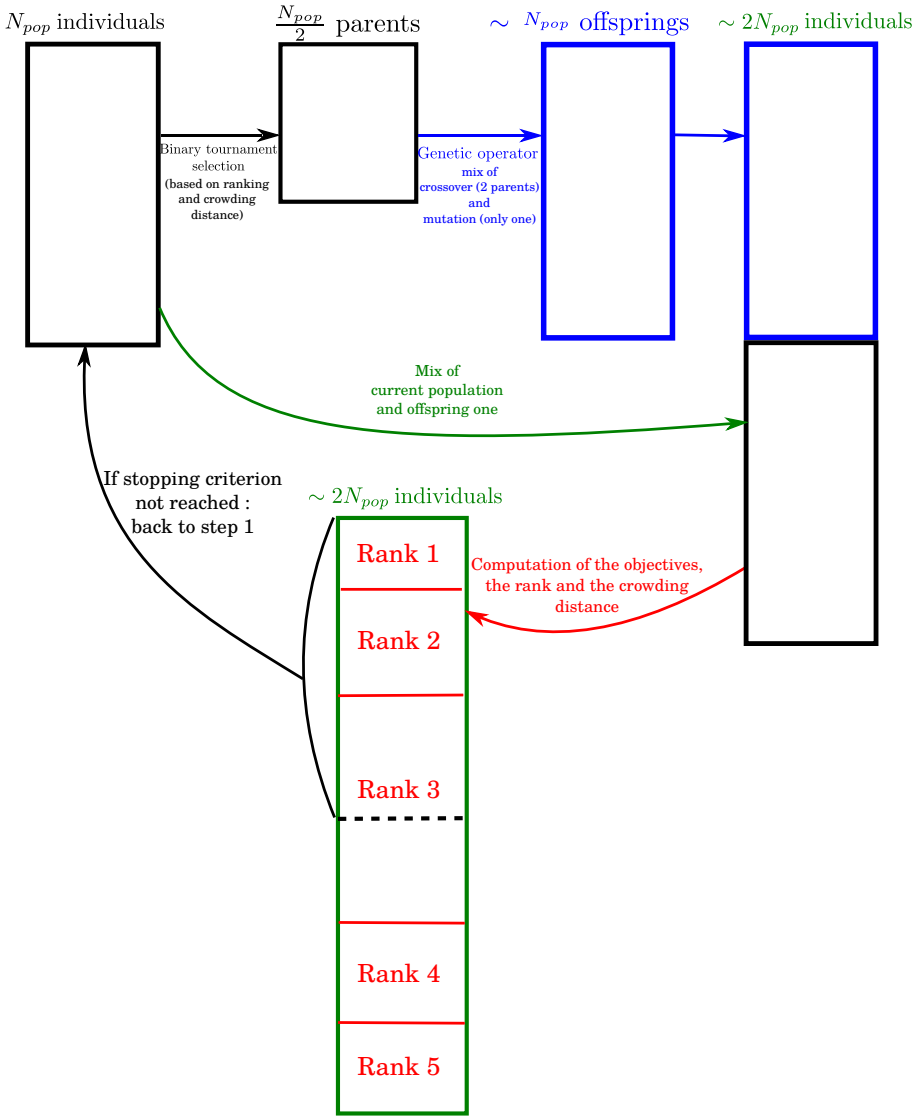


FIGURE 5.10 – Illustration du fonctionnement d'une génération de NSGA-II

il peut être nécessaire d'évaluer les individus de la population retournée par l'algorithme sur les fonctions objectif de référence.

Par exemple, l'association de MO-CMA-ES avec des modèles de substitution a été introduite par Loshchilov dans sa thèse. Il applique une méthode similaire à celle du mono-objectif [Loshchilov, 2013] décrite à la section 5.1.3. La différence ici provient du fait qu'il n'y a pas qu'un modèle de substitution mais autant que de fonctions objectif.

D'autres études mélangent des algorithmes évolutionnaires à des modèles de substitution. Par exemple, Akhtar utilise des métamodèles de type RBF dans son étude [Akhtar et Shoemaker, 2015]. En partant d'un plan d'expériences initial, un algorithme multi-objectif évolutionnaire est utilisé sur les prédictions du modèle RBF des objectifs afin de choisir les nouveaux points à rajouter au DOE. Pour cela, les solutions se situant dans les zones les moins peuplées de l'espace des objectifs sont choisies. Cela permet une recherche locale plus intensive dans la zone de cette solution. Une fois cette intensification

réalisée, le point obtenu est rajouté au DOE.

D'autres méthodes basées sur cette idée mais utilisant le krigeage existent. Par exemple, [Hamdaoui *et al.*, 2014] applique également un algorithme évolutionnaire sur les prédictions par krigeage des objectifs afin d'enrichir le DOE au cours de l'optimisation. En plus d'utiliser les informations fournies par l'algorithme évolutionnaire telles que la distance de nichage, la variance de la prédiction fournie par le krigeage est également mise à profit afin de rajouter le point du front prédit qui est le plus incertain. Gong et ses coauteurs proposent aussi dans leur étude [Gong *et al.*, 2015] d'appliquer des méthodes évolutionnaires à un métamodèle de type krigeage. Le DOE est cette fois-ci enrichi avec les points du front de Pareto obtenu par NSGA-II sur les prédictions par krigeage des objectifs qui ont la distance de nichage la plus élevée. Toutefois, ces méthodes ont le désavantage de ne pas utiliser pleinement les informations fournies par le modèle de krigeage, par exemple à l'aide d'un critère similaire à l'amélioration espérée.

5.2.2.2 Méthodes basées sur l'amélioration espérée

À la différence des méthodes présentées précédemment, le modèle de substitution sert ici à diriger l'optimisation et non plus seulement à diminuer le nombre d'appels. Ceci signifie que c'est le front de Pareto calculé sur les points du plan d'expériences qui est retourné ici et non plus une population prédite par le modèle de substitution.

La première manière d'utiliser l'amélioration espérée introduite dans la section 5.1.3 est de résoudre successivement des problèmes mono-objectifs. Des études se sont intéressées à ce mélange et exploitent le critère EI mono-objectif introduit à l'équation (5.4). De sorte à trouver le front de Pareto en résolvant des problèmes mono-objectifs, Knowles dans sa méthode ParEGO [Knowles, 2006] utilise la pondération de Tchebycheff écrite à l'équation (5.6) en faisant varier les poids a_i. Il est alors possible à chaque itération de rajouter le point maximisant l'EI correspondant à chacun des problèmes mono-objectifs, à la manière de l'algorithme 5.2. Une autre étude [Zhang *et al.*, 2010] mélange la pondération de Tchebycheff et le critère EI mono-objectif dans le but de trouver le front de Pareto. Elle repose sur l'utilisation de MOEA/D afin de générer les poids a_i au cours de l'optimisation et enrichit le DOE à chaque itération avec le maximum de l'EI des sous-problèmes mono-objectifs. D'autres études proposent également de mélanger les méthodes évolutionnaires et le critère d'amélioration espérée mono-objectif par la résolution de problèmes mono-objectifs intermédiaires pour enrichir le plan d'expériences comme celle d'Hussein et Deb [Hussein et Deb, 2016].

Enfin, une dernière idée consiste à élargir la notion d'amélioration espérée au cas multi-objectif. Pour cela, il faut repartir de la définition de ce critère : l'amélioration espérée consiste à quantifier l'espérance de l'amélioration par rapport au minimum courant que peut apporter un nouveau point, comme détaillé à l'équation (5.4).

Pour rappel en mono-objectif, l'amélioration espérée correspond à l'espérance de la distance entre le minimum courant et la valeur de la fonction au point étudié. En multi-objectif, cette idée se traduit par le calcul de l'espérance de la quantification d'un progrès par rapport au front courant. Toutefois, ce progrès peut être défini de plusieurs façons. Par exemple, on peut citer une mesure du progrès basée sur la distance euclidienne [Forrester et Keane, 2009] ou encore sur l'amélioration de l'hypervolume [Emmerich *et al.*, 2011]. Elles sont présentées dans le cas où les objectifs f_i sont prédits par un modèle de krigeage $Y_i(\boldsymbol{x})$ comme à l'équation (3.16) avec

$\boldsymbol{Y}(\boldsymbol{x}) = \left(Y_1(\boldsymbol{x}), \cdots, Y_{N_{obj}}(\boldsymbol{x})\right)$. On considère ici que les processus à l'origine des différentes fonctions objectif sont indépendants. C'est une hypothèse communément admise pour éviter la lourdeur de la prise en compte de corrélations entre les fonctions objectif. Svenson [Svenson, 2011] a étudié la prise en compte des dépendances et a montré qu'elle apporte peu par rapport à la lourdeur supplémentaire qu'elle nécessite. Voici le détail de ces deux techniques :

— Le premier critère proposé par Forrester et Keane [Forrester et Keane, 2009] consiste à chercher le point qui maximise l'espérance de la distance euclidienne minimum I_E avec le front de Pareto courant. Cela s'écrit de manière équivalente comme à l'équation (5.14).

$$\mathbb{I}\text{nfo}\,(\boldsymbol{x}) = \mathbb{E}\left[I_E(x)\right] = \mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}\right) d\left(\bar{\boldsymbol{y}}(\boldsymbol{x}), \boldsymbol{Y}^{PF}\right) \tag{5.14}$$

$\bar{\boldsymbol{y}}(\boldsymbol{x})$ correspond aux coordonnées du centroïde, calculé comme l'espérance de la position de l'objectif en dessous du front de Pareto. Ses composantes $\{\bar{y}_i\,(\boldsymbol{x})\}_{1 \leqslant i \leqslant N_{obj}}$ sont définies à l'équation (5.15).

$$\bar{y}_i\,(\boldsymbol{x}) = \frac{\mathbb{E}\left[Y_i(\boldsymbol{x}) \leqslant f_i(\boldsymbol{x}^{PF})\right]}{\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}\right)} \tag{5.15}$$

Ce critère est représenté sur la figure 5.11.

— Le second critère est basé sur l'amélioration de l'hypervolume apportée par l'ajout d'un point $\boldsymbol{x}$. Il est donc nécessaire de définir cet hypervolume. Il se calcule à partir d'un point de référence $\mathcal{R}$ à définir par l'utilisateur. Ce point permet de déterminer le sous-espace $\mathbb{H}_{\mathcal{R}} = \left\{\boldsymbol{y} \in \mathbb{R}^{N_{obj}} \;:\; \boldsymbol{y} \prec \mathcal{R}\right\}$ dans lequel l'hypervolume est calculé :

$$\left\{\begin{array}{rcl} \mathcal{H} & = & \left\{\boldsymbol{y} \in \mathbb{H}_{\mathcal{R}} \;:\; \exists \boldsymbol{x}' \in \boldsymbol{x}^{PF},\; \boldsymbol{F}(\boldsymbol{x}') \prec \boldsymbol{y}\right\} \\ \mathcal{H}_x & = & \left\{\boldsymbol{y} \in \mathbb{H}_{\mathcal{R}} \;:\; \exists \boldsymbol{x}' \in \left\{\boldsymbol{x}^{PF}, \boldsymbol{x}\right\},\; \boldsymbol{F}(\boldsymbol{x}') \prec \boldsymbol{y}\right\} \end{array}\right.$$

Ces quantités sont représentées sur la figure 5.12. On remarque que le point $\mathcal{R}$ doit être choisi de sorte qu'il soit dominé par tous les points du front, ce qui signifie qu'il doit avoir au minimum les coordonnées du maximum sur chaque composante des points de $PF^{\star}$. Ceci permet donc de définir le progrès apporté par un point $\boldsymbol{x}$ en terme d'hypervolume $I_h(\boldsymbol{x}) = |\mathcal{H}_x - \mathcal{H}|$. Ici, $|\cdot|$ désigne la mesure de Lebesgue du volume d'un sous-espace borné de $\mathbb{R}^{N_{obj}}$. Ces définitions permettent finalement de définir l'amélioration espérée basée sur l'hypervolume, écrite à l'équation (5.16).

$$\mathcal{I}\text{nfo}\,(\boldsymbol{x}) = \mathbb{E}\,(I_h) = \int I_h(\boldsymbol{x}) \varphi_{\hat{\boldsymbol{F}}(\boldsymbol{x}), \hat{\boldsymbol{\sigma}}(\boldsymbol{x})}\,(y_1, y_2)\, dy_1 dy_2 \tag{5.16}$$

Un exemple d'application de ce critère pour la résolution de problèmes industriels se trouve dans la thèse de Binois [Binois, 2015]. Il a en particulier généralisé l'utilisation de l'amélioration espérée multi-objectif au cas où le nombre de paramètre de design total peut être important mais où seulement quelques uns d'entre eux sont influents. Une bibliothèque R a d'ailleurs été développée dans le cadre de cette thèse pour l'optimisation multi-objectif assistée par processus gaussiens : GPareto [Binois et Picheny, 2015].

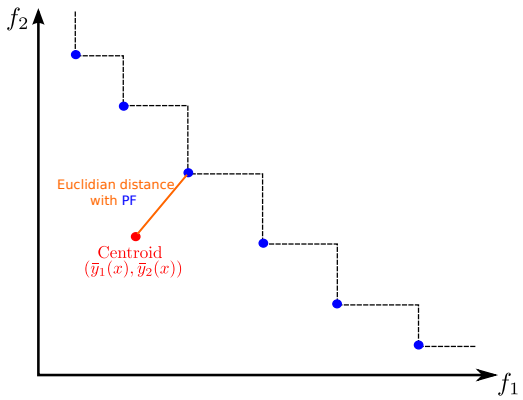


FIGURE 5.11 – Illustration du critère basé sur la distance euclidienne

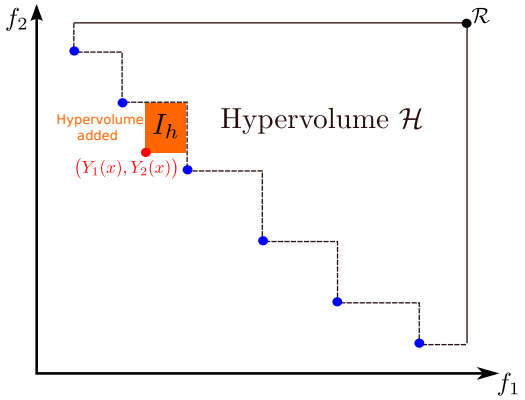


FIGURE 5.12 – Illustration du critère basé sur l'hypervolume

Le détail de leur calcul analytique dans le cas de deux fonctions objectif est développé en annexe B. Dans le cas de plus de deux objectifs, on peut par exemple citer l'étude de Hupkens [Hupkens *et al.*, 2014] qui propose de calculer l'intégrale de l'équation (5.16) en décomposant l'espace des objectifs en cellules, permettant par exemple d'obtenir une complexité de l'ordre de $O(N_{PF}^3)$ avec trois objectifs. Enfin, au-delà de trois objectifs, ces critères ne sont plus aussi facilement calculables analytiquement et peuvent nécessiter une estimation par des méthodes Monte-Carlo. Féliot utilise par exemple dans son étude [Féliot *et al.*, 2016] les méthodes du type Monte-Carlo séquentiel basées sur l'algorithme de Metropolis-Hastings afin de calculer et d'optimiser le critère hypervolume. L'utilisation d'algorithmes de type DIRECT ou évolution différentielle est également répandue pour maximiser ces critères.

En ce qui concerne les avantages et les inconvénients des deux formulations du critère EI multi-objectif introduites ici, l'hypervolume a l'avantage d'être plus robuste sur un grand nombre de cas tests comme cela a été montré par Parr [Parr, 2013]. De plus, Wagner et ses coauteurs [Wagner *et al.*, 2010] ont montré que le critère basé sur l'hypervolume avait de meilleures propriétés mathématiques. Il a toutefois pour désavantage d'obliger l'utilisateur à définir un point de référence $\mathcal{R}$ qui peut influer sur le résultat, notamment à cause des effets d'échelle que cela peut engendrer. En outre, le critère basé sur la distance euclidienne a pour avantage d'être moins sensible aux différences d'échelle entre les différents objectifs et d'être une valeur facilement interprétable.

Par ailleurs, d'autres critères d'enrichissement du DOE pour la résolution de problèmes multi-objectifs existent. Par exemple, Picheny utilise un critère de type SUR, déjà introduit à la section 4.2.3. Dans son étude [Picheny, 2015], la quantité dont il souhaite diminuer la variance d'estimation par l'ajout d'un point est l'hypervolume optimal recherché. Toutefois, ce critère est plus coûteux à estimer que l'amélioration espérée et peut nécessiter l'utilisation de deux échantillonnages Monte-Carlo imbriqués.

Ajout de plusieurs points à chaque itération afin de distribuer le calcul des objectifs : les critères d'amélioration espérée ne sont optimaux que pour l'ajout d'un seul point à chaque enrichissement du plan d'expériences. Il n'est donc pas possible de distribuer les appels aux fonctions objectif sans modifier le critère ou l'algorithme. Afin de pouvoir enrichir le plan d'expériences de plusieurs points simultanément, Parr [Parr, 2013]

modifie la phase de calcul du critère EI de sorte à garantir que ces points améliorent la connaissance de différentes zones du front. Pour cela, il ne calcule ces critères que dans la zone de l'espace des objectifs qui domine un *goal point* $\boldsymbol{y}^g \in \mathbb{R}^{N_{obj}}$. Ce point, représenté en vert sur le graphique 5.13, doit être choisi selon l'endroit de l'espace des objectifs que l'on souhaite enrichir. Plus précisément, ceci permet de limiter le calcul de l'intégrale dans les bornes fournies par ce point et représentées en rose sur le schéma.

$$\mathcal{I}\text{nfo}(\boldsymbol{x}) = \mathbb{E}\left(I \cap \left\{\left(Y_1(\boldsymbol{x}), \cdots, Y_{N_{obj}}(\boldsymbol{x})\right) \prec \boldsymbol{y}^g\right\}\right) \tag{5.17}$$

I peut être indifféremment la version euclidienne I_E ou hypervolume I_h. De cette manière,

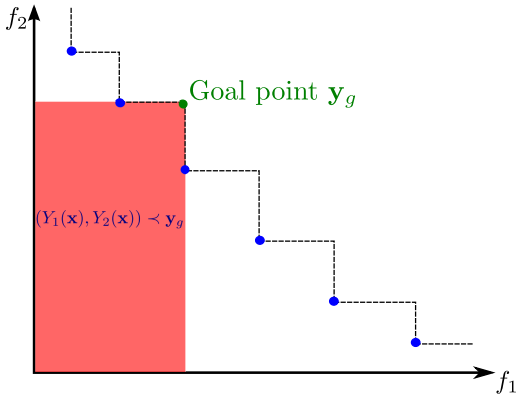


FIGURE 5.13 – Illustration du critère EI multi-points *goal-based*

il est possible de rajouter autant de points à chaque itération que ce qu'il y a de points dans le front de Pareto à l'itération courante.

C'est l'une des seules techniques existantes dans le cas de l'EI multi-objectif. Dans le cas d'un unique objectif, on peut citer l'étude de Ginsbourger et ses coauteurs [Ginsbourger *et al.*, 2010] qui traduisent mathématiquement ce que signifie l'ajout simultané de plusieurs points au niveau du calcul de l'espérance. Ceci lui permet de définir un critère théoriquement optimal. Toutefois, cette méthode n'a pas été adaptée au cas multi-objectif à notre connaissance et même si tel était le cas, elle ne permettrait pas de garantir un enrichissement du plan d'expériences dans différentes zones du front bien distribuées sur l'espace des objectifs (ce que permet de faire la méthode proposée par Parr).

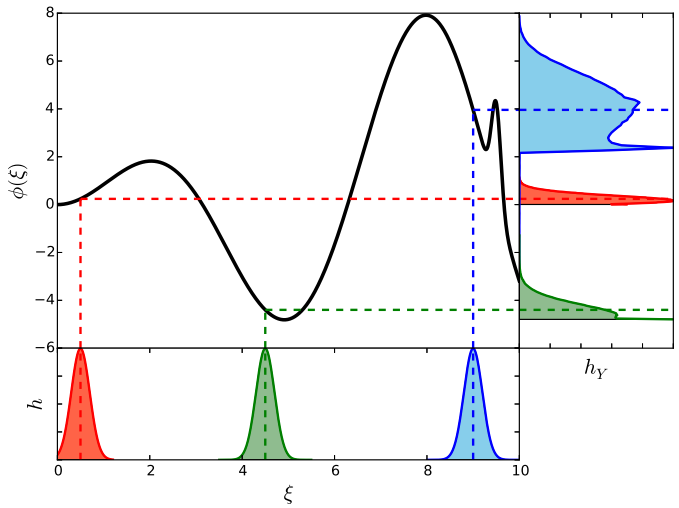


FIGURE 5.14 – Illustration de l'effet d'incertitudes en entrée d'un modèle déterministe pour l'optimisation

5.3 Prise en compte des incertitudes en optimisation

Sachant que le problème d'optimisation à résoudre est incertain, il est nécessaire de détailler ce que proposent les travaux existants dans la prise en compte d'incertitudes sur les paramètres d'entrée d'un problème d'optimisation, aussi bien dans le cas mono-objectif que multi-objectif. Pour prendre en compte ces incertitudes, nous nous intéressons particulièrement aux mesures de risque définies au chapitre 4. Pour rappel, ces mesures sont :

— l'α-quantile q_α
— l'α-superquantile Q_α
— le pire cas $\max_\xi(f_1(x,\xi))$

Nous pouvons les classer en deux types. Le premier type concerne les mesures probabilistes, c'est-à-dire qui prennent en compte la loi de probabilité des entrées afin d'être calculées. C'est le cas de l'α-quantile et de l'α-superquantile. Le deuxième type concerne les mesures déterministes qui sont celles qui ne prennent pas en compte la densité de probabilité mais traitent les incertitudes comme des intervalles de variation déterministes. Le pire cas en est un exemple.

Les incertitudes sont notées $\boldsymbol{\xi} = (\boldsymbol{\xi}_x, \boldsymbol{\xi}_e)$ et peuvent porter aussi bien sur les paramètres de design $\boldsymbol{x}$: $\boldsymbol{\xi}_x = (\xi_{x,1}, \cdots, \xi_{x,N_x})$, qu'environnementaux $\boldsymbol{\xi}_e = (\xi_{e,1}, \cdots, \xi_{e,N_e})$. Les paramètres environnementaux représentent des paramètres à fixer lors d'une optimisation déterministe afin de calculer la fonction objectif. Comme expliqué au chapitre précédent, il est parfois délicat de fixer certains paramètres. Traiter le manque de connaissance comme une incertitude permet d'éviter de trouver un optimum irréaliste et irréalisable.

Dans le cas de l'optimisation, une incertitude sur le paramètre d'entrée rend la sortie du modèle $f(\boldsymbol{x}, \boldsymbol{\xi})$ incertaine, même si la fonction f est déterministe. Ceci est illustré sur la figure 5.14. Cette figure montre notamment que la densité de la sortie est très variable, même en dimension 1 et avec des incertitudes suivant une loi normale tronquée en entrée. Ceci justifie le fait d'utiliser des méthodes de propagation d'incertitudes élaborées. En effet, la forme variable de la densité de la sortie ne permet pas de faire d'hypothèses pertinentes

sur cette densité. Par ailleurs, la figure 5.14 illustre également le fait que dans le cas général, la sortie de l'espérance des incertitudes en entrée ne coïncide pas avec l'espérance de la densité en sortie. Or, c'est l'approximation qui est faite lorsque le paramètre est traité de manière déterministe et que l'on néglige les incertitudes. Ceci justifie donc l'intérêt de prendre en compte les incertitudes dès la définition d'un problème d'optimisation.

Une branche de l'optimisation active et qui permet de prendre en compte les incertitudes portant sur le modèle est l'optimisation stochastique ou *stochastic optimization* [Spall, 2005]. Elle consiste à rechercher les paramètres de contrôle $\boldsymbol{x}^\star$ qui minimisent un modèle f dont la mesure est bruitée. C'est-à-dire que l'utilisateur n'a accès qu'à une version bruitée de ce modèle $\mathcal{F}(\boldsymbol{x}) = f(\boldsymbol{x}) + \varepsilon(\boldsymbol{x})$ où ε est la partie stochastique qui correspond au bruit. Par exemple, ceci peut être le cas d'un code stochastique qui ne retourne pas une valeur déterministe à une entrée donnée $\boldsymbol{x}$ fixée. Il existe des méthodes basées sur l'échantillonnage préférentiel [Zhao et Zhang, 2014] (présenté au chapitre 4) permettant de résoudre ce genre de problèmes. Néanmoins, même si le cas où les entrées sont incertaines peut être traité de cette manière, ces méthodes ne sont pas les plus adaptées puisqu'elles ont pour but de filtrer le bruit plutôt que de chercher à minimiser le risque qu'il peut induire.

Dans le cas où l'on considère que la fonction est déterministe (donc non bruitée) mais que les incertitudes portent uniquement sur les paramètres de design et environnementaux, l'optimisation sous incertitudes a d'abord consisté à résoudre des problèmes du type :

$$\min_{\boldsymbol{x}\in\mathcal{D}_x} \mathbb{E}_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right]$$

Ce qui peut s'interpréter comme un filtrage des incertitudes en calculant l'effet moyen de l'aléa sur la sortie, une idée proche de l'optimisation stochastique. Cette formulation est répandue puisqu'elle permet de considérer les effets des incertitudes à un coût raisonnable. Par exemple, l'étude de Janusevskis et Le Riche [Janusevskis et Le Riche, 2013] propose de coupler la minimisation et l'estimation de la mesure de robustesse à l'aide d'un modèle de substitution de f de type krigeage : ce modèle de krigeage lui permet d'approcher $\mathbb{E}_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right]$ par un processus gaussien puis de maximiser à l'aide de ce dernier processus un critère de type amélioration espérée (présenté à la section 5.1.3.1) pour choisir le point $(\boldsymbol{x}^\star,\boldsymbol{\xi}^\star)$ permettant d'enrichir le DOE.

De sorte à rajouter une information sur la variabilité de la sortie par rapport aux entrées, d'autres travaux considèrent aussi le calcul de la variance de la sortie [Taguchi et Phadke, 1989] :

- soit en formulant le problème à l'aide de deux objectifs :
  $\boldsymbol{F}(\boldsymbol{x}) = \left(\mathbb{E}_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right], \mathrm{Var}_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right]\right)$.
- soit en se ramenant à un problème mono-objectif combinant l'espérance et la variance : $\mathbb{E}_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right] + k\sqrt{\mathrm{Var}_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right]}$.

Toutefois, l'espérance et la variance ne permettent que de prendre en compte les effets moyens des incertitudes sur la sortie. De ce fait, ce ne sont pas des mesures de risque, comme cela a été détaillé au chapitre 4. De plus, la version mono-objectif présente le désavantage de ne pas avoir une interprétation facile par rapport à la densité de la sortie. Malgré cela, elle reste une manière envisageable de pénaliser les solutions à trop forte variabilité.

À présent, concernant les mesures de risques qui nous intéressent, il existe surtout des études permettant de résoudre une optimisation de type $\min_{\boldsymbol{x}}\max_{\boldsymbol{\xi}}$ aussi appelée

pire cas. Toutefois, ceci signifie qu'elles ne prennent pas en compte la densité de probabilité des entrées et ne considèrent les incertitudes que comme des intervalles de variation. En ce qui concerne l'utilisation de mesures de risque probabilistes en optimisation, le domaine existant dans la littérature s'en approchant le plus est l'optimisation sous contraintes fiabilistes aussi appelée *Reliability Based Design Optimization* (RBDO) [Haldar et Mahadevan, 2000]. Elle a pour but de résoudre des problèmes de la forme :

$$\begin{cases} \min_{\boldsymbol{x}\in\mathcal{D}_x} f(\boldsymbol{x}) \\ \text{s.c } \mathbb{P}\left(g_i(\boldsymbol{x},\boldsymbol{\xi}) \leqslant 0\right) \geqslant g_i^{tol} \ , \ 1 \leqslant i \leqslant N_{constr} \end{cases} \tag{5.18}$$

où g_i correspond à des fonctions contraintes dépendant des incertitudes et g_i^{tol} est la probabilité de tolérance d'une défaillance. Il est également possible de minimiser l'espérance de f plutôt que f de sorte à prendre en compte les incertitudes sur la fonction objectif en plus de la contrainte, comme expliqué dans l'étude de [Schuëller et Jensen, 2008]. Une référence récente et intéressante sur le sujet puisque combinant l'optimisation par modèles de substitution et optimisation fiabiliste est la thèse de Bichon [Bichon, 2010]. D'autres études proposent d'utiliser des modèles de substitution afin de propager l'incertitude, par exemple à l'aide de chaos polynomial [Suryawanshi et Ghosh, 2016]. Cependant, l'inconvénient de la RBDO est qu'elle nécessite de traduire l'aversion au risque par des contraintes. Cela n'est pas équivalent à la minimisation des mesures de risque visée ici.

De ce fait, la formulation du problème industriel correspond davantage à la formulation pire cas. Nous détaillons à présent les méthodes de résolution de ces problèmes proposées dans la littérature. Nous présentons d'abord les techniques qui peuvent s'appliquer à n'importe quelle mesure de risque présentée puisqu'elle consiste à découpler la propagation d'incertitudes et l'optimisation. La seconde partie a pour but de détailler les techniques permettant de coupler l'estimation de la mesure de robustesse et l'optimisation. Uniquement le cas du pire cas est détaillé dans cette partie puisque c'est le seul présent dans la littérature.

5.3.1 Résolution par découplage de l'estimation de la mesure de risque et de l'optimisation

Dans le cas général, une manière de résoudre un problème d'optimisation sous incertitudes consiste en une résolution séparée de la propagation d'incertitudes et de l'optimisation. En d'autres termes, cela signifie qu'on applique un algorithme d'optimisation classique à la sortie d'un estimateur de la mesure de robustesse tel que présenté au chapitre précédent.

Par exemple, [Ong *et al.*, 2006] utilise un algorithme évolutionnaire afin de résoudre la partie optimisation $\min_{\boldsymbol{x}} \tilde{f}(\boldsymbol{x})$ et utilise un algorithme de région de confiance afin de résoudre le pire cas pour chaque individu $\boldsymbol{x}$, c'est-à-dire $\tilde{f}(\boldsymbol{x}) = \max_{\xi\in\mathcal{D}_\xi} f(\boldsymbol{x},\boldsymbol{\xi})$.

Cette décomposition du problème en deux étapes peut impliquer un nombre d'appel important à la fonction déterministe f, du fait du coût d'estimation important d'une mesure de risque. De ce fait, certaines méthodes intègrent dans l'optimisation la prise en compte de la robustesse.

5.3.2 Résolution par couplage de l'estimation de la mesure de risque et de l'optimisation

Il existe peu de travaux dans la littérature couplant la phase d'optimisation et l'estimation de la mesure de risque du chapitre 4. De ce fait, on s'intéresse dans cette partie aux études qui ont proposé ce couplage dans la résolution d'un problème d'optimisation pire cas de la forme suivante :

$$\min_{\boldsymbol{x}\in\mathcal{D}_x} \max_{\boldsymbol{\xi}\in\mathcal{D}_\xi} f(\boldsymbol{x},\boldsymbol{\xi}) \tag{5.19}$$

5.3.2.1 Optimisation pire cas par relaxation

Pour résoudre le problème de l'équation (5.19), l'idée de la relaxation est de ne pas travailler sur l'ensemble continu $\mathcal{D}_\xi$ pour les incertitudes mais de se restreindre à un ensemble fini d'évènements $\mathcal{D}_\xi^d$, que l'on enrichit à chaque itération. Pour initialiser l'algorithme, ils définissent cet ensemble discret à l'aide d'un point $\boldsymbol{\xi}^0$, $\mathcal{D}_\xi^d = \left\{\boldsymbol{\xi}^0\right\}$ qui peut être un point tiré aléatoirement. Ensuite, la technique de relaxation consiste à itérativement résoudre des problèmes mono-objectifs et à rajouter le point $\boldsymbol{\xi}^\star$ solution de $\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi^d} f(\boldsymbol{x}^\star,\boldsymbol{\xi})$ à l'ensemble discret $\mathcal{D}_\xi^d$ à chaque itération. La relaxation est notamment détaillée par Marzat et ses coauteurs dans leur étude [Marzat *et al.*, 2012]. L'algorithme 5.3 résume les étapes de recherche par relaxation.

Algorithme 5.3 Algorithme d'optimisation pire cas par relaxation

Input : la fonction à minimiser f
Les ensembles de définition $\mathcal{D}_x$ et $\mathcal{D}_\xi$
Tolérance ε
Output : La solution de l'équation (5.19)

- $\mathcal{D}_\xi^d = \left\{\boldsymbol{\xi}^0\right\}$ tiré aléatoirement dans $\mathcal{D}_{\boldsymbol{\xi}}$;
- Calculer $\boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathcal{D}_x}{\operatorname{argmin}} \left[\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi^d} f(\boldsymbol{x},\boldsymbol{\xi})\right]$;
- Poser $f^\star = \max_{\boldsymbol{\xi}\in\mathcal{D}_\xi^d} f(\boldsymbol{x}^\star,\boldsymbol{\xi})$;
- Calculer $\boldsymbol{\xi}^1 = \underset{\boldsymbol{\xi}\in\mathcal{D}_\xi}{\operatorname{argmax}} f(\boldsymbol{x}^\star,\boldsymbol{\xi})$ et poser $k=1$;

while $f(\boldsymbol{x}^\star,\boldsymbol{\xi}^k) - f^\star \leqslant \varepsilon$ **do**

- $\mathcal{D}_\xi^d = \left\{\mathcal{D}_\xi^d, \boldsymbol{\xi}^k\right\}$;
- Calculer $\boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathcal{D}_x}{\operatorname{argmin}} \left[\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi^d} f(\boldsymbol{x},\boldsymbol{\xi})\right]$;
- Poser $f^\star = \max_{\boldsymbol{\xi}\in\mathcal{D}_\xi^d} f(\boldsymbol{x}^\star,\boldsymbol{\xi})$ puis $k = k+1$;
- Calculer $\boldsymbol{\xi}^k = \underset{\boldsymbol{\xi}\in\mathcal{D}_\xi}{\operatorname{argmax}} f(\boldsymbol{x}^\star,\boldsymbol{\xi})$;

end

- Retourner $\boldsymbol{x}^\star$;

L'algorithme s'arrête dès que le rajout d'un nouveau point dans l'ensemble discret ne permet pas de trouver un meilleur minimum à une tolérance donnée ε. Il est illustré dans

l'étude de Marzat et de ses coauteurs que le nombre d'itération nécessaire est petit et reste de l'ordre de $k = 5$.

Assistance par modèle de substitution : toutes les étapes de recherche d'optimum dans un espace continu peuvent être réalisées par EGO détaillé à l'algorithme 5.2 comme détaillé dans l'étude [Marzat *et al.*, 2012].

5.3.2.2 Optimisation pire cas par algorithmes coévolutionnaires

Une autre manière de résoudre des problèmes d'optimisation sous incertitudes de type pire cas et basée sur les algorithmes évolutionnaires est donnée par les méthodes coévolutionnaires [Paredis, 1995]. Elles reposent sur l'utilisation de deux populations évoluant simultanément $\mathcal{P}_x$ et $\mathcal{P}_\xi$. La première population $\boldsymbol{x} \in \mathcal{P}_x$ permet de minimiser la partie déterministe à $\boldsymbol{\xi}$ fixé (choisi dans la population $\mathcal{P}_\xi$). La deuxième population $\boldsymbol{\xi} \in \mathcal{P}_\xi$ permet de maximiser la partie incertaine à $\boldsymbol{x}$ fixé (pris dans la population $\mathcal{P}_x$). Ceci signifie que les deux populations évoluent l'une après l'autre, de manière décorrélée. L'idée introduite par la coévolution ressemble donc à l'idée de la relaxation en se ramenant à des ensembles discrets pour résoudre la partie incertaine. C'est notamment ce que réalise Krohling et ses coauteurs dans l'étude [Krohling *et al.*, 2004]. L'algorithme évolutionnaire qui y est utilisé est l'algorithme à essaims particulaires aussi appelé *Particle Swarm Optimization* et introduit dans [Kennedy et Eberhart, 1995].

Une autre manière d'utiliser les deux populations disponibles est d'utiliser la population d'individus de la partie incertaine de sorte à tester les individus de la population déterministe $\mathcal{P}_x$. Ceci permet ainsi de modifier l'opérateur de sélection de sorte à favoriser les points de l'espace de design dont le pire cas est minimum. C'est notamment ce que propose Branke dans son étude [Branke et Rosenbusch, 2008]. Branke dans une autre étude [Branke *et al.*, 2013] a même redéfini la dominance en utilisant cette population de test dans le cas d'un problème multi-objectif où tous les objectifs sont des pire cas :

$$\min_{\boldsymbol{x} \in \mathcal{D}_x} \left[\max_{\boldsymbol{\xi} \in \mathcal{D}_\xi} f_1(\boldsymbol{x}), \cdots, \max_{\boldsymbol{\xi} \in \mathcal{D}_\xi} f_{N_{obj}}(\boldsymbol{x}) \right] \tag{5.20}$$

L'inconvénient de cette méthode est qu'elle est adaptée au cas où le domaine incertain est discret et de cardinal fini, il correspond à différents scénari parmi lesquels on cherche le pire cas. De ce fait, il est très difficile dans le cas où l'espace $\mathcal{D}_\xi$ est continu de choisir des scénarios pour composer la population $\mathcal{P}_\xi$.

5.3.2.3 Couplage par équilibre de Nash

L'étude de Lung et Dimitrescu [Lung et Dumitrescu, 2011] propose d'utiliser l'équilibre de Nash afin de résoudre l'équation (5.19). En effet, ce problème peut être traduit en un jeu à somme nulle composé de deux joueurs. Le premier joueur a une fonction de gain donnée par $f(\boldsymbol{x}, \boldsymbol{\xi})$ et le deuxième joueur a une fonction de gain donnée par $-f(\boldsymbol{x}, \boldsymbol{\xi})$. L'équilibre est ensuite trouvé en utilisant un algorithme à évolution différentielle, introduit en section 5.1.2.4.

5.3.2.4 Couplage par modèle de substitution

Une autre manière de résoudre une optimisation pire cas en utilisant un modèle de substitution et s'inspirant également de EGO est proposée par [Rehman *et al.*, 2014]. Elle consiste à utiliser le fait que le point où se réalise un maximum peut être connu. À chaque itération, Rehman ajoute au plan d'expériences le point $\boldsymbol{x}$ où la valeur de $\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi} f(\boldsymbol{x},\boldsymbol{\xi})$ a le plus de chance d'améliorer la connaissance de la solution de 5.19. Pour cela, il maximise l'amélioration espérée du point $\left(\boldsymbol{x},\hat{\boldsymbol{\xi}}^\star\right)$ où se réalise le maximum sur le métamodèle $\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi} \hat{f}(\boldsymbol{x},\boldsymbol{\xi})$ autour de chaque point de design $\boldsymbol{x}$. Pour calculer l'amélioration espérée, la valeur de la fonction en ce point est comparée à celle de $\min_{\boldsymbol{u}\in\mathcal{D}_x}\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi}\hat{f}(\boldsymbol{u},\boldsymbol{\xi})$ qui est la solution de 5.19 sur le métamodèle. Ceci est détaillé à l'algorithme 5.4.

Algorithme 5.4 Algorithme d'optimisation pire cas assisté par modèle de substitution

Input : la fonction à minimiser f
Les ensembles de définition $\mathcal{D}_x$ et $\mathcal{D}_\xi$

Output : La solution de l'équation (5.19)

· À partir d'un DOE initial tiré par LHS, construire $\hat{f}(\boldsymbol{x},\boldsymbol{\xi})$;

while *Critère d'arrêt non satisfait* **do**

· Calculer $\min_{\boldsymbol{x}\in\mathcal{D}_x}\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi}\hat{f}(\boldsymbol{x},\boldsymbol{\xi})$;

· Pour chaque $\boldsymbol{x}$ calculer le point $\hat{\boldsymbol{\xi}}^\star$ tel que
$\hat{y}_{max}(\boldsymbol{x}) = \hat{f}(\boldsymbol{x},\hat{\boldsymbol{\xi}}^\star) = \max_{\boldsymbol{\xi}\in\mathcal{D}_\xi}\hat{f}(\boldsymbol{x},\boldsymbol{\xi})$ et $\hat{\sigma}_{max}(\boldsymbol{x}) = \hat{\sigma}\left(\boldsymbol{x},\hat{\boldsymbol{\xi}}^\star\right)$;

· Calculer $\boldsymbol{x}^\star$ maximisant :

$$\mathcal{I}\text{nfo}(\boldsymbol{x}) = I(\boldsymbol{x})\Phi\left[\frac{I(\boldsymbol{x})}{\hat{\sigma}_{max}(\boldsymbol{x})}\right] + \hat{\sigma}_{max}(\boldsymbol{x})\varphi_{(0,1)}\left[\frac{I(\boldsymbol{x})}{\hat{\sigma}_{max}(\boldsymbol{x})}\right]$$

Avec $I(\boldsymbol{x}) = \min_{\boldsymbol{u}\in\mathcal{D}_x}\{\hat{y}_{max}(\boldsymbol{u})\} - \hat{y}_{max}(\boldsymbol{x})$;

· Rajouter le point $\left(\boldsymbol{x}^\star,\hat{\boldsymbol{\xi}}^\star\right)$ au DOE et reconstruire $\hat{f}$;

end

· Retourner $\min_{\boldsymbol{x}\in\mathcal{D}_x}\max_{\boldsymbol{\xi}\in\mathcal{D}_\xi}\hat{f}(\boldsymbol{x},\boldsymbol{\xi})$ et le point où se réalise l'optimum;

Cet algorithme a pour avantage de discriminer en peu d'appels à la fonction f les zones où la solution a peu de chances de se trouver. Toutefois, elle présente le désavantage de chercher le minimum sur le métamodèle et de retourner la solution évaluée uniquement sur ce modèle de substitution. On n'a donc pas l'assurance que la solution trouvée soit la solution de l'équation (5.19) portant sur la vraie fonction f.

Troisième partie

Développement

Chapitre 6

Optimisation Multi-objectif

Sommaire

L'objectif de cette partie est de proposer une résolution de problème multi-objectif (MOO pour *Multi-Objective Optimization*) requérant peu d'évaluations pour un temps de restitution raisonnable. Les fonctions objectif peuvent être coûteuses à évaluer. En effet, le cas test industriel présenté à l'équation (2.12) nécessite de savoir résoudre le problème suivant avec $\boldsymbol{x} \in \mathcal{D}_x \subset \mathbb{R}^{N_x}$:

$$\begin{cases} \min\limits_{x} f_1(\boldsymbol{x}) \\ \min\limits_{x} f_2(\boldsymbol{x}) \end{cases} \tag{6.1}$$

Or l'une des fonctions objectif est une mesure de robustesse prenant en compte les incertitudes. En d'autres termes, $f_1(\boldsymbol{x}) = \rho_\xi [f(\boldsymbol{x}, \boldsymbol{\xi}_x, \boldsymbol{\xi}_e)]$. Ceci signifie que le calcul de f_1 peut être coûteux, même dans le cas où la fonction f a un temps d'exécution raisonnable, par exemple de l'ordre de la minute ou de la dizaine de minutes.

Afin de proposer un algorithme MOO parcimonieux, nous clarifions d'abord le cadre d'étude ainsi que les cas tests utilisés. Ils nous permettent de comparer différentes méthodes et d'évaluer leur efficacité en présence de fonctions objectif très coûteuses en temps de calcul. Plus précisément, deux algorithmes MOO de la littérature sont étudiés dans ce

cadre. Ceci permet ainsi de conclure sur les bénéfices et les inconvénients de chacune de ces deux méthodes. La première méthode étudiée est NSGA-II qui est un algorithme évolutionnaire qui a fait ses preuves ([Deb *et al.*, 2002],[Deb, 2011]). La seconde méthode, appelé ici MOEGO pour EGO Multi-Objectif, est basée sur les modèles de substitution et sur l'utilisation du critère d'amélioration espérée multi-objectif. Cette utilisation d'un modèle de substitution permet en particulier de réduire le nombre d'appels aux fonctions coût. À la différence de l'étude de Féliot [Féliot *et al.*, 2016], nous souhaitons distribuer les appels aux fonctions objectif, ce qui nécessite des modifications de l'algorithme MOEGO classique.

Enfin, à l'aide des conclusions tirées sur les deux précédentes algorithmes, la dernière étape consiste à proposer et valider un algorithme MOO basé sur le critère d'amélioration espérée et l'algorithme génétique NSGA-II. Cet algorithme composite permet de bénéficier des avantages de chacune de ces méthodes.

6.1 Présentation du cadre d'étude et des cas tests utilisés

6.1.1 Cadre d'étude

L'objectif de cette partie est de résoudre le problème détaillé à l'équation (6.1) dans le cas où les deux fonctions objectif sont coûteuses. Pour rappel et comme expliqué à la partie 5.3, la façon la plus intuitive de résoudre un problème du type de l'équation (2.12) est d'utiliser un algorithme d'optimisation classique que l'on applique à une mesure de robustesse de la (ou des) fonction(s) coût incertaine(s). Ceci signifie que l'évaluation de l'une des fonctions objectif, par exemple $f_1(\boldsymbol{x}) = \rho_\xi [f(\boldsymbol{x}, \boldsymbol{\xi}_x, \boldsymbol{\xi}_e)]$, nécessite l'estimation de la mesure de robustesse ρ. Or, cette estimation peut être gourmande en nombre d'appels à la fonction f. De ce fait, même dans le cas où cette fonction a un temps d'exécution qui peut paraître réduit, par exemple de l'ordre de la minute, l'objectif incertain $f_1(\boldsymbol{x})$ peut être considéré comme coûteux.

Les ingénieurs attendent de l'optimisation qu'elle mette en évidence des solutions innovantes, c'est-à-dire auxquelles ils n'auraient pas pensé puisque peu répandues dans le domaine de l'aéronautique par exemple. De ce fait, ils ne souhaitent pas contraindre la méthode et recherchent tous les compromis optimaux, en d'autres termes le front de Pareto optimal complet. Par conséquent, les méthodes traduisant un problème multi-objectif en un problème mono-objectif avec ou sans contraintes ne sont pas envisageables puisqu'elles ne fournissent qu'une unique solution dépendante de l'*a priori* utilisé (cf section 5.2). De plus, du fait que l'optimisation est utilisée afin de trouver des solutions innovantes, il est important de ne pas rater de compromis optimaux. En d'autres termes, ceci signifie que le front de Pareto optimal doit être approché assez finement.

Pour résumer, la résolution de l'équation (2.12) comporte des objectifs qui ont des temps d'exécution qui peuvent être importants. En outre, il est nécessaire de ne rater aucune solution et ainsi de fournir une approximation du front de Pareto suffisamment précise. Dès lors, il est nécessaire de choisir un algorithme MOO permettant :

1. peu d'évaluations. Ceci signifie qu'il doit conduire à un front de Pareto en peu d'appels distribués (de l'ordre de 5 à 10 N_x). C'est la raison pour laquelle les méthodes basées sur les modèles de substitution sont ici intéressantes.
2. un temps de restitution raisonnable. Pour cela, il est important de pouvoir distribuer les appels aux objectifs de sorte à réduire le temps de restitution total de

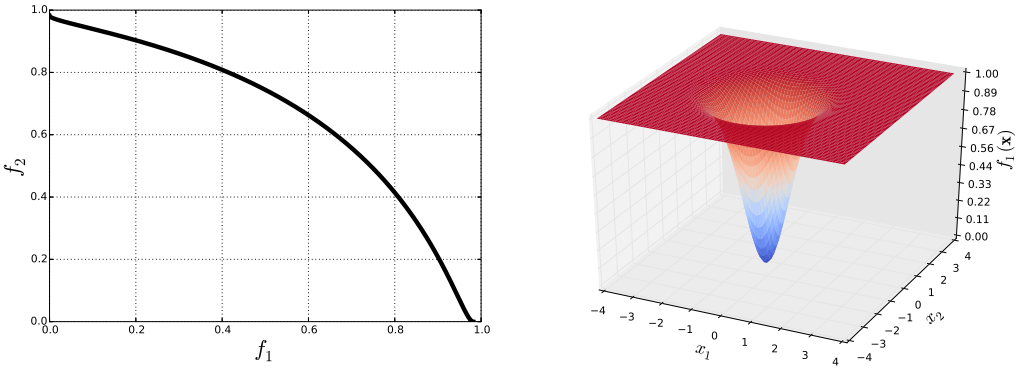


FIGURE 6.1 – Illustration du front de Pareto optimal et de l'un des objectifs avec $N_x = 2$ pour le cas test MOP2

l'algorithme. Le nombre de processeurs disponibles pour ce travail est de l'ordre de 10.

3. une discrétisation régulière du front de Pareto complet.

6.1.2 Cas tests et mesure d'erreur utilisés

Cas tests analytiques : Afin de comparer les diverses méthodes d'optimisation, deux cas tests analytiques de dimension modifiable et conduisant à des fronts de Pareto de forme atypique sont utilisés. Le premier est le cas test appelé MOP2 provenant de l'étude de Fonseca et Fleming et cité dans [Coello *et al.*, 2013].

$$-4 \leqslant x_i \leqslant 4 \text{ et } \begin{cases} f_1(\boldsymbol{x}) = 1 - \exp\left[-\sum_{i=1}^{N_x}\left(x_i - \frac{1}{\sqrt{N_x}}\right)^2\right] \\ f_2(\boldsymbol{x}) = 1 - \exp\left[-\sum_{i=1}^{N_x}\left(x_i + \frac{1}{\sqrt{N_x}}\right)^2\right] \end{cases} \tag{6.2}$$

On déduit de l'équation (6.2) que les objectifs sont fortement non-linéaires. De plus, la figure 6.1 montre que cette fonction peut être difficile à approcher. En effet, si l'on échantillonne grossièrement, la fonction peut avoir l'apparence d'une fonction constante égale à 1. Enfin, la forme du front de Pareto optimal est concave, ce qui peut exclure l'utilisation de certaines méthodes adaptées à des fronts de forme convexe. Les points Pareto-optimaux sont l'ensemble des $\boldsymbol{x}$ appartenant à $PF^\star$ défini tel que :

$$PF^\star = \left\{ \boldsymbol{F}(\boldsymbol{x}) \text{ avec } \boldsymbol{x} = (t, \cdots, t) \text{ et } t \in \left[-\frac{1}{\sqrt{N_x}}, \frac{1}{\sqrt{N_x}}\right] \right\} \tag{6.3}$$

Le second cas test est le cas ZDT3 qui est notamment cité dans l'étude de [Zitzler *et al.*, 2000]. Celui-ci est également non-linéaire et comme le montre la figure 6.2, le front de Pareto optimal est discontinu et convexe par morceaux.

$$0 \leqslant x_i \leqslant 1 \text{ et } \begin{cases} f_1(\boldsymbol{x}) = g(\boldsymbol{x})[1.0 - \sqrt{\frac{x_1}{g(\boldsymbol{x})}} - \frac{x_1}{g(\boldsymbol{x})}\sin(10\pi x_1)] \\ f_2(\boldsymbol{x}) = x_1 \end{cases} \tag{6.4}$$

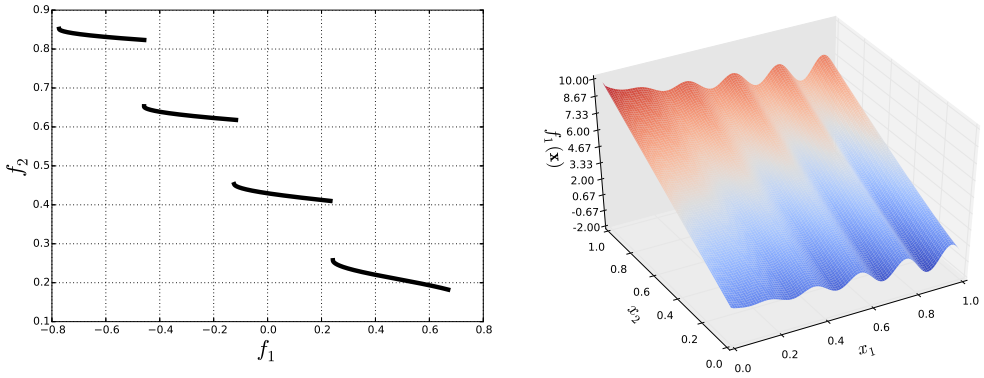


FIGURE 6.2 – Illustration du front de Pareto optimal et de l'un des objectifs avec $N_x = 2$ pour le cas test ZDT3

$$g(\boldsymbol{x}) = 1 + \frac{9}{N_x - 1} \sum_{i=2}^{N_x} x_i$$

Le front de Pareto optimal théorique est donné par l'équation suivante :

$$\left\{ \begin{array}{rcl} PF^\star & = & \left\{ (t_1, t_2) \ : \ t_2 \in T \text{ et } t_1 = 1 - \sqrt{t_2} - t_2 \sin\left(10\pi t_2\right) \right\} \\ T & = & [0, 0.083] \cup [0.1822, 0.258] \cup [0.1822, 0.258] \\ & & \cup [0.409, 0.454] \cup [0.618, 0.653] \cup [0.823, 0.852] \end{array} \right. \tag{6.5}$$

Cette discontinuité permet en particulier d'éprouver les algorithmes MOO étant donné que cela peut complexifier la recherche de l'intégralité du front de Pareto.

Mesure d'erreur : pour pouvoir comparer les différents algorithmes MOO et pour en mesurer la convergence, il est nécessaire de choisir une mesure d'erreur. Dans le cas mono-objectif, la distance relative de la solution retournée par un algorithme à la vraie valeur du minimum de la fonction est souvent utilisée. En multi-objectif, cela peut être plus compliqué puisqu'il ne suffit pas seulement que les solutions soient sur le vrai front de Pareto du problème pour que l'algorithme soit satisfaisant. Il faut également que les solutions retournées permettent de discrétiser finement ce front. Dans la suite, on appelle Front de Pareto Optimal ($PF^\star$) le front théoriquement solution du problème MOO. Dans le cas des problèmes multi-objectifs présentés ici, ils sont connus analytiquement. Le front de Pareto retourné par l'algorithme étudié est appelé Front de Pareto Approché (PF).

Une mesure communément utilisée pour mesurer la convergence en multi-objectif est l'hypervolume, détaillée dans la section 5.2.2.2 de l'état de l'art. Pour rappel, l'hypervolume correspond au volume de l'espace des objectifs dominé par le PF retourné par l'algorithme. Elle permet donc de vérifier que le volume dominé par le PF et celui dominé par le $PF^\star$ sont comparables. Cependant, elle ne permet pas de mesurer l'espacement moyen entre les points du PF retourné par l'algorithme. Ils peuvent de ce fait être mal répartis le long du front. De plus, le calcul de ce volume ne peut se faire qu'en définissant un point de référence $\mathcal{R}$. La valeur obtenue de l'hypervolume est donc dépendante de ce point de référence.

Pour s'affranchir de ces problèmes, il est possible d'utiliser la formule analytique du $PF^\star$ dans l'espace des objectifs et de calculer une distance relative entre ce $PF^\star$ et le

PF retourné par l'algorithme testé. Ceci revient à généraliser la mesure utilisée en mono-objectif au cas multi-objectif. Toutefois, afin de vérifier que les solutions du PF $\boldsymbol{Y}^{PF}$ sont à la fois Pareto-optimales et bien distribuées sur le front, il peut être intéressant de discrétiser le $PF^\star$ dans l'espace des objectifs : $\boldsymbol{Y}^{PF^\star}$. En discrétisant le $PF^\star$ avec la discrétisation visée (déterminée par le nombre de points souhaités dans ce front $N_{PF^\star}$), la mesure de convergence suivante peut être proposée :

$$\mathcal{M}\left(PF, PF^\star\right) = \frac{1}{N_{PF^\star}} \sum_{i=1}^{N_{PF^\star}} \min_{\boldsymbol{y} \in Y_{PF}} d(y^{PF_i^\star}, \boldsymbol{y}) \tag{6.6}$$

C'est donc la distance moyenne des points du $PF^\star$ discrétisé $\boldsymbol{Y}^{PF^\star}$ avec le PF retourné par l'algorithme étudié $\boldsymbol{Y}^{PF}$ qui est utilisée, comme illustrée sur la figure 6.3. En choisissant une discrétisation satisfaisante, ceci garantit de favoriser les fronts retournés qui sont au moins aussi bien discrétisés que le front optimal.

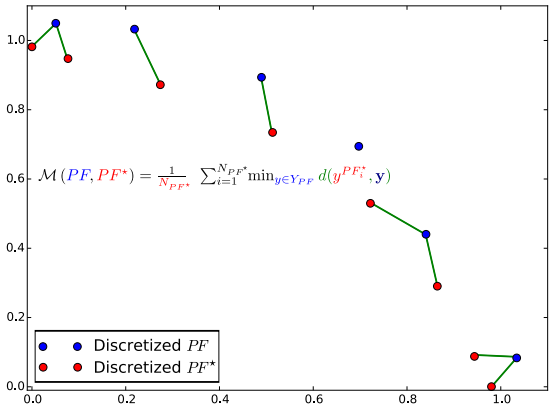


FIGURE 6.3 – Illustration de la mesure $\mathcal{M}$ entre le front de Pareto optimal $PF^\star$ et le front de Pareto approché PF

Cette mesure est par exemple utilisée par Zhang et Li dans leur étude [Zhang et Li, 2007] afin de quantifier la discrétisation régulière de $PF^\star$ par le front PF retourné par l'algorithme évolutionnaire multi-objectif MOEA/D. Ce critère peut se trouver sous le nom de *Inverted Generational Distance* (IGD) dans la plupart des études existantes.

6.2 Intérêts et limites de l'algorithme NSGA-II

De sorte à avoir une référence à laquelle se comparer et dont la robustesse n'est plus à prouver, cette partie a pour but de présenter les résultats de l'algorithme NSGA-II, détaillé en section 5.2, sur les deux cas tests. L'objectif est de mesurer la capacité de convergence de cet algorithme lorsque nous avons un budget fixé. Pour cela, nous quantifions le nombre d'appels nécessaires aux fonctions coûteuses pour atteindre une distance cible entre le front approché par l'algorithme et le front optimal.

Sachant que les objectifs sont coûteux et qu'il est envisagé par la suite d'intégrer un modèle de substitution à cet algorithme, NSGA-II a entièrement été développé en Python. Ceci permet ainsi de librement insérer l'utilisation d'un modèle de substitution à n'importe quelle de ses étapes.

Pour rappel, cet algorithme s'inspire de la biologie : il consiste à initialiser une population de N_{pop} individus de manière aléatoire, puis à les faire évoluer selon des règles provenant de la sélection naturelle. Cela présente l'avantage de combiner une exploration intensive, à l'aide des opérateurs de croisement et de mutation, à la garantie de non-régression que permet l'opérateur de sélection basé sur la distance de nichage et le rang. Ce mélange d'exploration et de non-régression permet d'assurer la convergence. Toutefois, son principal inconvénient est qu'il requiert un nombre d'appels important aux fonctions objectif afin de converger vers un PF de qualité acceptable.

Pour illustrer cela, l'algorithme NSGA-II a été appliqué sur les cas tests analytiques introduits (MOP2 et ZDT3). Les paramètres sont choisis de sorte à ne pas faire exploser le nombre d'appels aux deux fonctions objectif : une population de taille 50 et des paramètres de mutation et de croisement mélangeant intensification et exploration. Pour cela, nous utilisons un paramètre de mutation $c_{mut} = 4$ et un paramètre de croisement $c_{cross} = 2$. Comme illustrées sur les figures 5.8 et 5.9, ces valeurs permettent de ne pas faire un choix trop prononcé entre intensification de la recherche en une zone et trop grande exploration. NSGA-II est un algorithme stochastique, il est donc nécessaire de répéter l'optimisation de nombreuses fois pour en vérifier la convergence. Dans notre étude, nous considérons une répétition de 50 optimisations. Enfin, on se donne deux critères d'arrêt, le premier porte sur la variation de l'hypervolume (mesure détaillée dans la partie 5.2.2.2) entre la génération précédente et la génération courante :

$$\frac{\left|\mathcal{H}^{gen} - \mathcal{H}^{gen-1}\right|}{\mathcal{H}^{gen-1}} \leqslant \varepsilon \tag{6.7}$$

Dans les calculs lancés ici, $\varepsilon = 10^{-4}$ et le point de référence est le point $\mathcal{R} = (1.1, 1.1)$ pour MOP2 et le point $\mathcal{R} = (8.0, 1.1)$ pour ZDT3. La figure 6.4 illustre le nombre d'appels aux fonctions objectif une fois ce critère d'arrêt atteint. On remarque qu'il est supérieur à 1000 pratiquement dans tous les cas.

Cependant, ce graphe ne donne pas d'informations sur la qualité du front en fonction du nombre d'appels réalisés aux fonctions objectif.

Le second critère d'arrêt, plus précis mais nécessitant la connaissance du $PF^\star$, est le suivant :

$$\mathcal{M}\left(PF, PF^\star\right) \leqslant \mathcal{M}^{target} \tag{6.8}$$

Pour calculer cette distance $\mathcal{M}$ (introduite à l'équation (6.6)), nous choisissons de discrétiser le front de Pareto optimal $PF^\star$ avec 200 points dans les deux cas. Nous fixons ensuite la distance cible à $\mathcal{M}^{target} = 0.0013$ pour MOP2 et $\mathcal{M}^{target} = 0.005$ pour ZDT3. Une

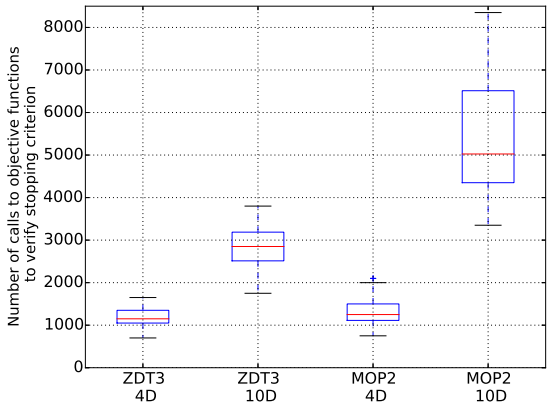


FIGURE 6.4 – Boîtes à moustaches du nombre d'appels aux fonctions objectif une fois le critère d'arrêt atteint pour ZDT3 et MOP2 en dimension 4 et 10

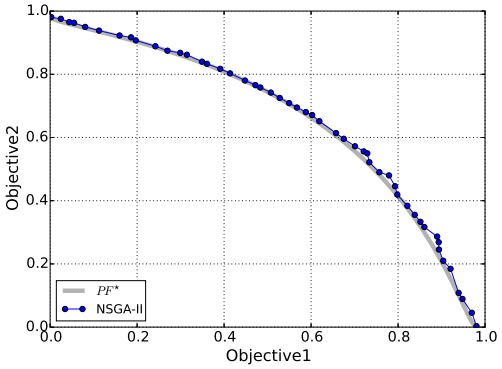


FIGURE 6.5 – Illustration d'un front de Pareto Approché à une distance du $PF^\star$ égale à la distance cible pour le cas test MOP2

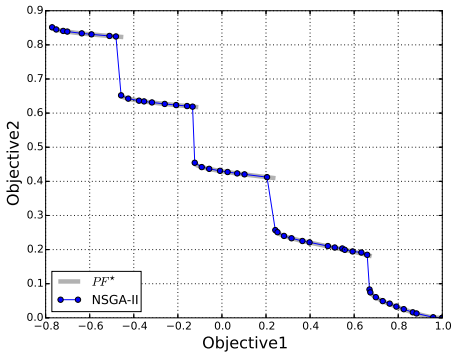


FIGURE 6.6 – Illustration d'un front de Pareto Approché à une distance du $PF^\star$ égale à la distance cible pour le cas test ZDT3

figure illustrant la distance à $PF^\star$ du PF en fonction du nombre d'appels aux fonctions objectif permet d'extraire le budget de calcul nécessaire à l'obtention d'un PF suffisamment discrétisé. Les dimensions qui sont testées sont la dimension 4 (cf figures 6.7 et 6.9) et la dimension 10 (cf figures 6.8 et 6.10). La dimension 4 coïncide avec celle du problème d'optimisation industriel. La dimension 10 est intéressante puisqu'elle correspond à une limite pour les modèles de substitution de type krigeage, notamment du fait du fléau de la dimension, notamment illustré dans l'introduction du livre [Hastie *et al.*, 2009]. Ces courbes illustrent la décroissance de la moyenne de cette distance sur les 50 optimisations avec le nombre d'appels aux fonctions objectif. La ligne noire sur ces figures représente la distance cible définie plus haut. Comme l'illustrent les figures 6.5 et 6.6, cette distance cible correspond à des fronts satisfaisants. Les courbes des figures 6.7 à 6.10 permettent donc de déduire le nombre moyen d'appels nécessaires aux fonctions objectif afin de converger. Le tableau 6.1 résume ce résultat dans le cas où l'on s'intéresse au nombre d'itérations à partir duquel la moyenne de la distance atteint la distance cible sur les 50 optimisations.

Jusqu'à présent, on ne s'est servi que de la valeur moyenne sur les cinquante optimisations pour en déduire le budget nécessaire à un niveau de convergence satisfaisant. Pour

MOP2 en dimension 4 :

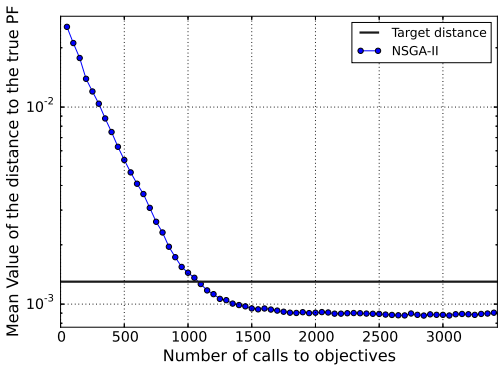


FIGURE 6.7 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 4

MOP2 en dimension 10 :

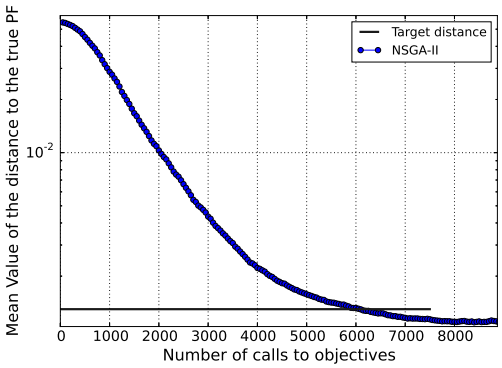


FIGURE 6.8 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 10

ZDT3 en dimension 4 :

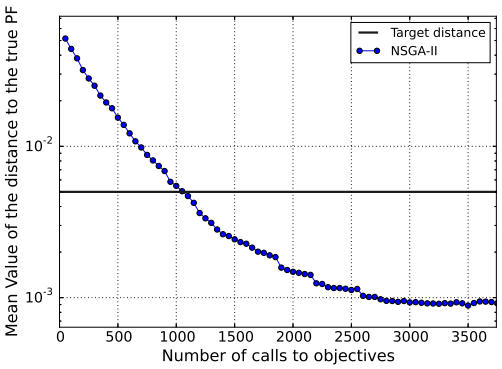


FIGURE 6.9 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 4

ZDT3 en dimension 10 :

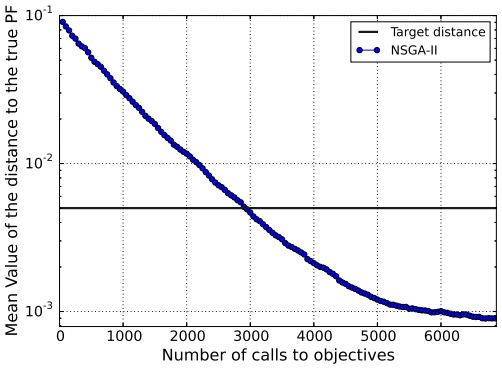


FIGURE 6.10 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 10

MOP2 :

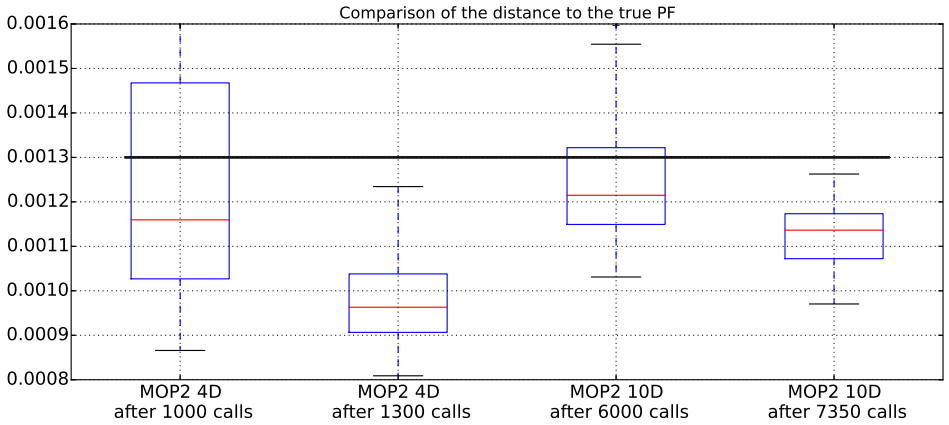


FIGURE 6.11 – Graphique représentant les boîtes à moustache de la distance entre le front approché et le front optimal pour le cas test MOP2

ZDT3 :

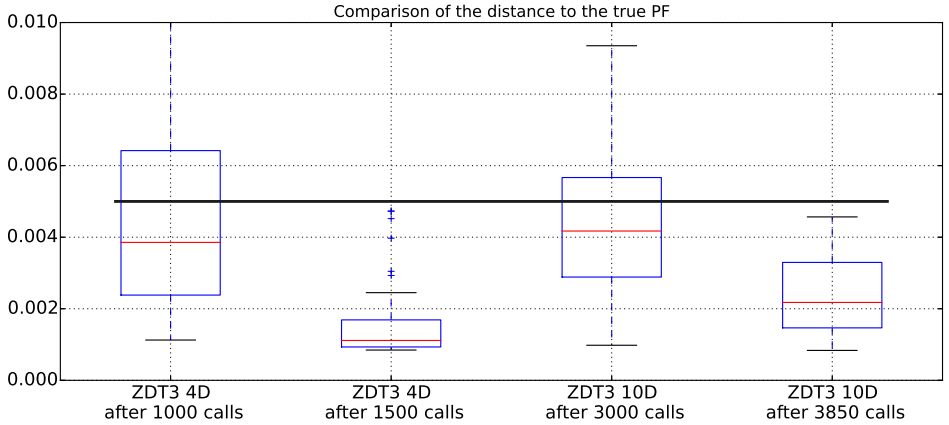


FIGURE 6.12 – Graphique représentant les boîtes à moustache de la distance entre le front approché et le front optimal pour le cas test ZDT3

montrer la robustesse de l'algorithme NSGA-II, il faut donc mesurer le nombre d'appels nécessaires aux fonctions objectif pour que les cinquante optimisations donnent une précision acceptable. Pour cela, les boîtes à moustaches du niveau de convergence à nombre d'appels aux fonctions objectif fixé sont fournies sur les figures 6.11 pour MOP2 et 6.12 pour ZDT3.

	Dimension 4	Dimension 10
MOP2	1000	6000
ZDT3	1000	3000

TABLEAU 6.1 – Tableau récapitulant le nombre d'appels nécessaires afin que la valeur moyenne sur 50 optimisations de la distance atteigne la distance cible

	Dimension 4	Dimension 10
MOP2	1300	7350
ZDT3	1500	3850

TABLEAU 6.2 – Tableau récapitulant le nombre d'appels nécessaires afin que la valeur maximum sur 50 optimisations de la distance atteigne la distance cible

Le tableau 6.2 récapitule le nombre d'appels nécessaires afin que pour chacune des 50 optimisations la distance atteigne la distance cible. Ce que l'on remarque d'abord, c'est que le nombre d'appels où la moyenne de la distance passe en dessous du seuil espéré (représenté sur le tableau 6.1) ne correspond pas au nombre d'appels où le maximum de la distance est plus petit que la distance cible. Ensuite, l'écart entre le nombre d'appel moyen et le nombre d'appel maximum nécessaires est de 20 générations (ce qui correspond à 1000 appels avec une population de 50 individus). Ceci montre que l'algorithme NSGA-II est robuste puisque toutes les optimisations convergent avec un nombre d'appels d'un ordre de grandeur équivalent.

Toutefois, le dernier tableau nous montre que ces algorithmes évolutionnaires ne peuvent être utilisés lorsque les fonctions objectif ont un temps d'exécution trop conséquent puisqu'il faut au minimum les appeler 1000 fois. Les algorithmes génétiques sont très facilement parallélisables. En effet, à chaque génération, N_{pop} calculs peuvent être réalisés simultanément. Or, lorsque l'on a la possibilité de distribuer les appels à la fonction sur un nombre N_{CPU} de CPU réduit, par exemple 10, le nombre d'appels distribués nécessaires resterait trop important pour converger de manière acceptable : entre 100 et 150 appels distribués au minimum ce qui est conséquent en dimension 4 seulement. Le budget maximal se situe plutôt entre $5N_x$ et $10N_x$ ce qui signifie entre 20 et 40 appels distribués en dimension 4. On peut néanmoins remarquer que si davantage de processeurs sont disponibles, l'algorithme NSGA-II peut alors devenir compétitif.

Pour résumer, l'algorithme NSGA-II présente l'avantage de converger dans tous les cas grâce à une exploration intensive de l'espace des entrées. Toutefois, le nombre d'appels aux fonctions objectif nécessaire peut rapidement être important. Les méthodes basées sur les modèles de substitution peuvent apporter une solution à ce problème.

6.3 Intérêts et limites des algorithmes de type amélioration espérée - MOEGO

Afin de diminuer le nombre d'appels aux fonctions objectif, une approche consiste à utiliser des modèles de substitution comme détaillé dans la partie 5.2.2. Les modèles de substitution sont définis dans le chapitre 3. Pour les construire, ils nécessitent la génération d'un plan d'expériences. Une idée couramment répandue dans la littérature et détaillée à l'algorithme 5.2 est de partir d'un plan d'expériences initial généré aléatoirement puis d'enrichir le plan d'expériences séquentiellement avec le point apportant le plus d'informations pour trouver le minimum d'une fonction.

Même s'il est possible de simplement appliquer l'algorithme NSGA-II sur la prédiction par un métamodèle et d'enrichir le plan d'expériences avec les points optimaux, Jones [Jones, 2001] a montré l'inefficience de ce genre de stratégie. La raison principale de cet échec est que le modèle de substitution n'est pas une représentation parfaite de la fonction qu'il substitue. En conséquence, et pour ne pas rater de solutions malgré l'utilisation d'un métamodèle, il est primordial d'utiliser une stratégie d'enrichissement de plan d'expériences qui utilise une estimation de l'erreur commise par le modèle de substitution.

C'est notamment ce que propose l'algorithme EGO Multi-Objectif (MOEGO) basé sur l'amélioration espérée multi-objectif. Pour rappel, l'amélioration espérée est un critère d'enrichissement du plan d'expériences d'un modèle de krigeage. Dans le cas multi-objectif, il est détaillé dans la partie 5.2.2. Il consiste à calculer l'espérance du progrès dans la connaissance du $PF^\star$. Pour réduire le temps de restitution de l'algorithme, nous mettons en place et testons la méthode proposée par Parr [Parr, 2013] pour distribuer les appels aux objectifs. Elle est détaillée dans la Section 5.2.2.

Le but de cette partie est donc de tester ces techniques sur des cas tests (MOP2 et ZDT3). L'application de ces méthodes à ces cas tests, en utilisant la mesure de succès introduite à l'équation (6.6), n'a pas été faite à notre connaissance. Cela nous permet de vérifier leurs avantages et leurs limites dans le cadre que l'on s'est fixé. D'autre part, il existe plusieurs manières de définir une amélioration espérée : l'une basée sur l'hypervolume et qui présente de meilleures propriétés mathématiques, notamment de continuité, l'autre basée sur la distance euclidienne et qui présente l'avantage de ne pas avoir à définir de point de référence $\mathcal{R}$. Les études existantes sur ce sujet et détaillées dans la partie 5.2.2 n'ont pas tranché clairement sur l'efficacité de ces deux critères, en particulier lorsqu'ils sont utilisés de manière distribuée. Ils sont donc comparés afin de choisir le plus adapté aux cas étudiés.

6.3.1 Mise en place de MOEGO avec calcul distribué

L'amélioration espérée est un critère d'enrichissement optimal dans le cas où un seul point est ajouté à chaque itération. Or, afin de réduire le temps de restitution de l'algorithme multi-objectif et pour explorer davantage l'espace des objectifs, il peut être intéressant de rajouter plusieurs points à chaque itération. Ceci peut en effet diminuer le temps d'exécution si les fonctions objectif peuvent être évaluées en plusieurs points simultanément. L'algorithme de la méthode MOEGO multi-points est détaillé sur la figure 6.13.

L'une des seules méthodes existantes dans la littérature permettant de rajouter plusieurs points au plan d'expériences dans des zones disjointes de l'espace des objectifs est

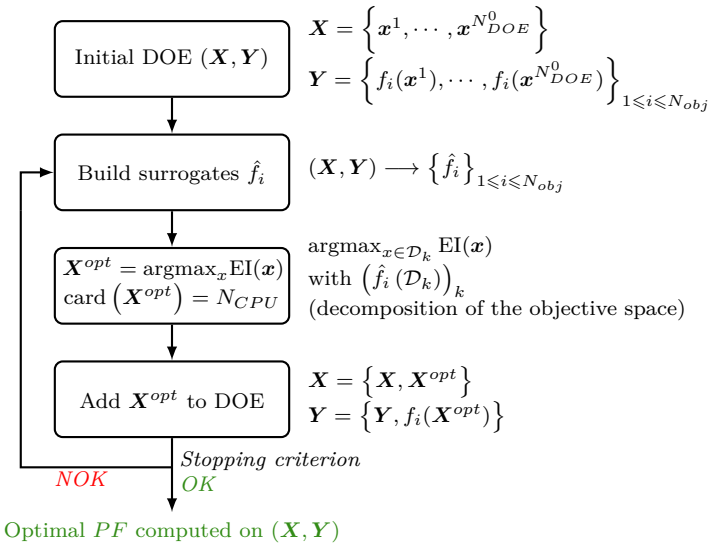


FIGURE 6.13 – Illustration de la méthode MOEGO multi-points par la formule de Parr

proposée par Parr [Parr, 2013]. Elle repose sur le calcul du critère d'amélioration espérée dans une zone réduite de l'espace des objectifs. Afin de choisir plusieurs points à rajouter simultanément, l'idée est de localiser la maximisation de l'amélioration espérée dans des zones proches des points du front courant $\boldsymbol{Y}^{PF} = \left(\boldsymbol{y}^{PF_1}, \cdots, \boldsymbol{y}^{PF_{N_{PF}}}\right)$. Plus précisément et en se référant à la figure 5.13, l'amélioration espérée est uniquement calculée dans la zone de l'espace des objectifs qui domine des *goal points*, qui sont choisis comme les intersections entre les zones dominées par les points du front courant.

Après chaque rajout simultané de points, le modèle de krigeage de chacune des fonctions objectif est reconstruit. En pratique, le krigeage est mis en place à l'aide du *package* Scikit-learn codé en Python [Pedregosa *et al.*, 2011].

Choix des N_{CPU} *goal points* utilisés :

Soient $\boldsymbol{y}^{g_1}, \cdots, \boldsymbol{y}^{g_{N_{PF}+1}}$ les $N_{PF}+1$ *goal points* potentiels (représentés sur la figure 6.14). On remarque que les points extrêmes du front courant $\boldsymbol{Y}^{PF}$ sont considérés comme des *goal points*. Sachant qu'à chaque itération, on souhaite ajouter N_{CPU} points au plan d'expériences, il faut distinguer deux cas :

1. Si le nombre de *goal points* $N_{PF}+1 \leqslant N_{CPU}$, alors ils sont tous utilisés pour résoudre l'équation (5.17).
2. Sinon, un choix doit être réalisé parmi les $N_{PF}+1$ *goal points* : pour cela, on n'utilise que ceux dont la distance euclidienne avec les deux points du front courant les plus proches est la plus grande. En ordonnant les points du front courant selon l'un des deux objectifs, cette distance est calculée de la manière suivante :

$$d(i) = \begin{cases} \infty & \text{si } i = 1 \text{ ou si } i = N_{PF}+1 \\ d\left(\boldsymbol{y}^{PF_{i-1}}, \boldsymbol{y}^{PF_i}\right) & \text{sinon} \end{cases} \tag{6.9}$$

Comme pour la *crowding distance* dans le cas de NSGA-II, nous pouvons remarquer que cette distance est infinie au niveau des points extrémaux du PF. Les deux

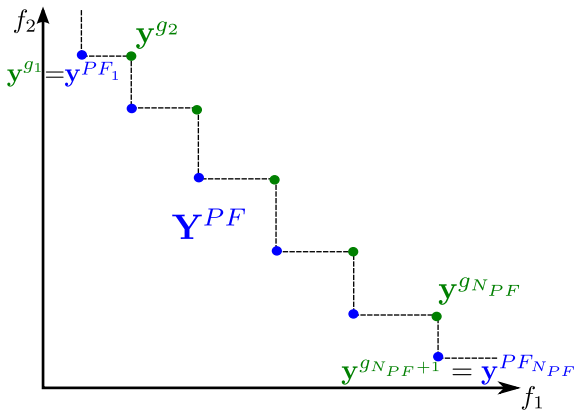


FIGURE 6.14 – Illustration de la position des *goal points* par rapport aux points du front courant représenté en bleu

extrémités du front sont donc forcément utilisées.

Pour résumer, il est donc nécessaire de résoudre N_{CPU} recherches de maximum de l'EI modifié à chaque étape d'enrichissement du plan d'expériences. Afin de choisir l'algorithme le plus adapté, il faut déterminer la forme mathématique de la fonction à optimiser. Dans le cas du critère EI basé sur la distance euclidienne, cette fonction objectif a pour inconvénient d'être non différentiable puisqu'elle est calculée comme la distance du centroïde au point du front le plus proche. Il y a donc des sauts de dérivée dans les zones où le centroïde change de point le plus proche dans le front courant. De ce fait, l'algorithme à évolution différentielle présenté en section 5.1.2.4 est un bon candidat puisqu'il permet de minimiser des fonctions sans hypothèses particulières sur la régularité de leur dérivée, alors que l'algorithme DIRECT, étant un algorithme lipschitzien, présuppose une certaine régularité de la dérivée.

6.3.2 Résultats sur les cas tests analytiques

Nous pouvons donc appliquer cet algorithme à nos deux cas tests. Comme pour NSGA-II dans la section précédente, il a été codé en Python. La *toolbox* scikit-learn [Pedregosa *et al.*, 2011] est utilisée pour construire le modèle de krigeage des objectifs, la *toolbox* Scipy [Jones *et al.*, 01] est utilisée pour l'algorithme à évolution différentielle permettant de maximiser l'amélioration espérée modifiée. Comme précédemment, nous répétons cinquante optimisations en dimension 4 et en dimension 10 aussi bien pour MOP2 que pour ZDT3. Le nombre de processeurs utilisés est fixé à 10. Enfin, la taille du plan d'expériences initial est de 10, de sorte à ne pas utiliser trop de budget dans l'initialisation et en consacrer une plus grande partie aux points maximisant le critère d'amélioration espérée. En effet, ce dernier tend naturellement à enrichir le plan d'expériences dans les zones où l'incertitude est trop importante, réalisant un compromis entre exploration et intensification.

Les résultats (figures 6.15 à 6.18) montrent l'évolution de la distance avec le nombre d'appels distribués aux objectifs. Ils permettent d'illustrer que la méthode basée sur le calcul du critère EGO multi-objectif a une capacité d'exploration limitée du front de Pareto entier. En effet, on remarque qu'en dehors du cas de MOP2 en dimension 4 sur la figure 6.15, aucune de ces méthodes ne parvient à converger rapidement vers la distance

visée. Dans le cas de ZDT3, on remarque même que ces méthodes ne convergent pas vers le $PF^\star$ et stagnent sur des solutions insatisfaisantes, quel que soit le nombre de dimensions (cf figures 6.17 en dimension 4 et 6.18 en dimension 10). La figure 6.19 illustre que MOEGO reste bloqué sur des solutions non Pareto-optimales. On remarque plus précisément en observant ces fronts que ce qu'il manque à ces méthodes, c'est l'exploration de l'espace des objectifs puisqu'une partie du front est bien capturée. Ceci est dû au fait que la méthode proposée par Parr consiste à limiter le calcul de l'amélioration espérée uniquement dans la zone de l'espace des objectifs dominée par des *goal points*. La recherche du maximum de cette amélioration espérée ne va donc pas favoriser l'exploration de l'espace des objectifs entiers du fait que le calcul de ce critère est ici trop contraint.

Ces figures permettent également de voir que sur les cas tests choisis, le critère basé sur la distance euclidienne donne de meilleurs résultats. Ceci s'explique par le fait que l'hypervolume n'a pas pour objectif de remplir le front de Pareto de sorte qu'il soit bien discrétisé, ce qui est pourtant ce que l'on recherche avec la mesure utilisée ici. Dans le cas général en effet, l'ajout de points dans certaines zones du front va peu améliorer la valeur de l'hypervolume. En d'autres termes, un front dont l'hypervolume est presque maximal ne sera pas forcément bien discrétisé alors que c'est ce que la mesure utilisée valorise ici. En particulier, dans le cas d'un front concave (comme MOP2) ou discontinu (comme ZDT3), peu de points peuvent suffire pour obtenir une valeur de l'hypervolume presque maximale. À l'inverse, la distance euclidienne détecte davantage les zones du front où il peut manquer des points : elle va tendre à combler les zones du front mal discrétisées puisque ce sont des zones où la distance euclidienne entre le point à rajouter et le front courant peut être importante. Enfin, c'est une mesure locale puisqu'elle est calculée à partir du point du front le plus proche alors que l'hypervolume est une mesure globale. Or, la localité de la distance euclidienne correspond davantage au découpage de l'espace des objectifs réalisé ici. Pour résumer, même si l'hypervolume est le meilleur critère possible pour l'ajout d'un point à chaque itération [Wagner *et al.*, 2010], cela n'est plus vrai lorsque l'on souhaite ajouter plusieurs points simultanément à divers endroits du front, comme c'est le cas ici.

Enfin, les figures 6.20 à 6.23 illustrent la robustesse de ces méthodes basées sur l'amélioration espérée. En effet, ces boîtes à moustaches représentent la valeur de la distance au front après un certain nombre d'itérations. Elles mettent donc en évidence la convergence rapide de toutes les optimisations réalisées, en particulier dans le cas du critère basé sur la distance euclidienne.

Pour résumer, les méthodes basées sur l'amélioration espérée multi-objectif permettent d'obtenir des solutions Pareto-optimales en un nombre d'appels distribués réduit. Toutefois, le manque d'exploration de l'espace des objectifs et de l'espace de design les empêche de capturer le front dans son intégralité, comme le montre la figure 6.19 sur le cas test ZDT3.

MOP2 en dimension 4 :

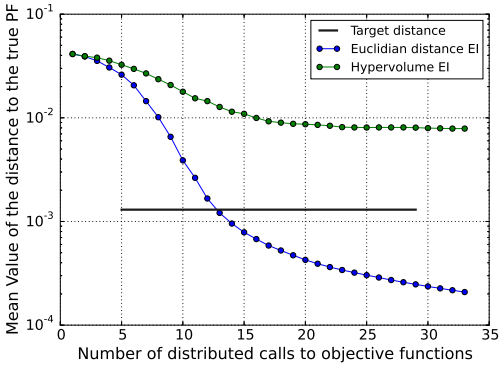


FIGURE 6.15 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 4

ZDT3 en dimension 4 :

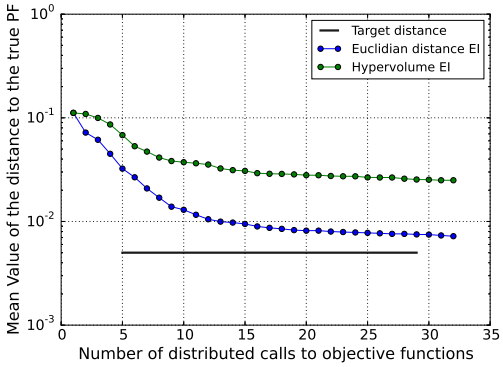


FIGURE 6.17 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 4

MOP2 en dimension 10 :

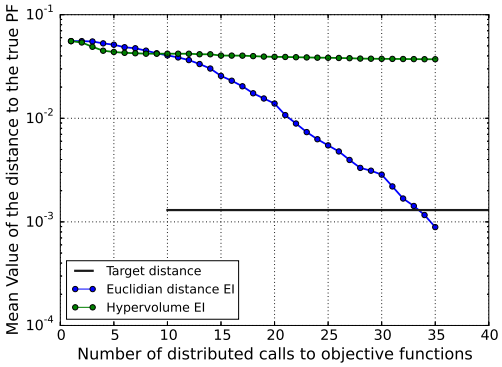


FIGURE 6.16 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 10

ZDT3 en dimension 10 :

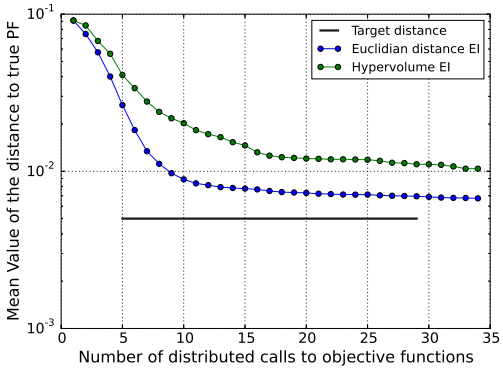


FIGURE 6.18 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 10

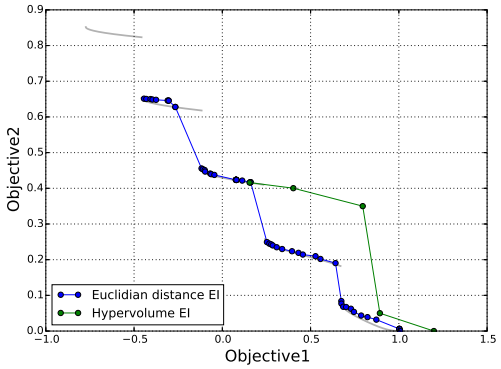


FIGURE 6.19 – Illustration d'un PF obtenu après 35 appels distribués pour le cas test ZDT3 en dimension 10

MOP2 en dimension 4 :

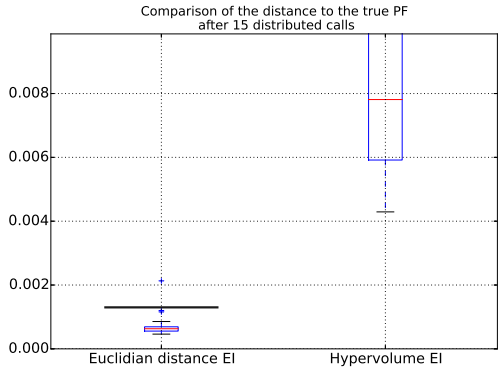


FIGURE 6.20 – Graphique représentant les boîtes à moustaches de la distance entre le front approché et le front optimal pour le cas test MOP2 en dimension 4

MOP2 en dimension 10 :

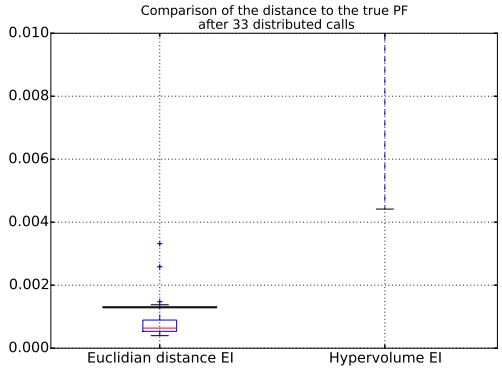


FIGURE 6.21 – Graphique représentant les boîtes à moustaches de la distance entre le front approché et le front optimal pour le cas test MOP2 en dimension 10

ZDT3 en dimension 4 :

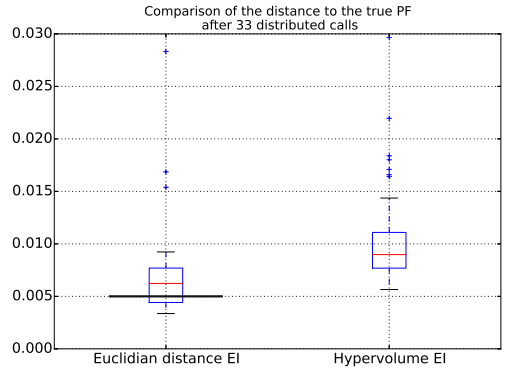


FIGURE 6.22 – Graphique représentant les boîtes à moustaches de la distance entre le front approché et le front optimal pour le cas test ZDT3 en dimension 4

ZDT3 en dimension 10 :

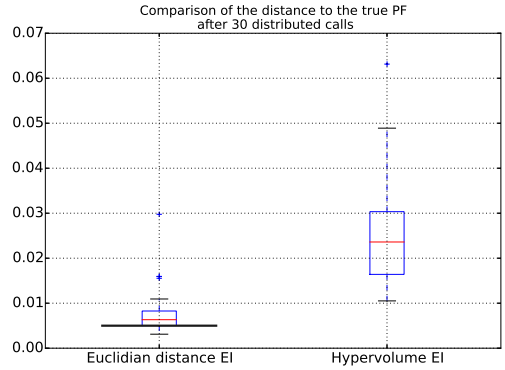


FIGURE 6.23 – Graphique représentant les boîtes à moustaches de la distance entre le front approché et le front optimal pour le cas test ZDT3 en dimension 10

6.4 MOEGO NSGA-II - une alternative pour la distribution des calculs

Nous avons donc mis en évidence que les algorithmes évolutionnaires sont robustes et explorent entièrement le $PF^\star$. Néanmoins, ils présentent le désavantage de nécessiter un grand nombre d'appels aux fonctions objectif. Par ailleurs, les méthodes basées sur l'amélioration espérée nécessitent peu d'appels aux fonctions objectif mais ne permettent pas d'obtenir l'intégralité du front, en particulier lorsque ce dernier a une forme particulière, comme MOP2 et ZDT3. L'idée de cette partie est donc de proposer puis de valider un algorithme d'optimisation conjuguant la robustesse de l'algorithme NSGA-II à la parcimonie permise par l'amélioration espérée.

La section 5.2.2.1 présente différentes façons de coupler modèles de substitution et algorithmes évolutionnaires. Le principal reproche que l'on puisse faire aux études présentées dans cette section est qu'elles n'utilisent pas toutes les informations fournies par le krigeage : elles utilisent soit l'écart type sur la prédiction fournie par l'estimateur, soit l'intervalle de confiance sur la prédiction que l'on peut également déduire de l'espérance et de l'écart type. Toutefois, il est détaillé à la section 5.1.3.1 qu'il peut être sous efficace de ne pas utiliser entièrement l'information fournie par le krigeage, c'est-à-dire la distribution de la prédiction $\boldsymbol{Y}(\boldsymbol{x})$ (cf équation (3.16)) en tout point $\boldsymbol{x} \in \mathcal{D}_x$.

C'est justement ce que permet de faire l'amélioration espérée. Toutefois, il a été montré à la section précédente que sur les cas étudiés, l'exploration de l'espace des objectifs n'est pas toujours suffisante et ne permet donc pas d'obtenir l'intégralité du front. L'utilisation de l'algorithme évolutionnaire peut permettre de compenser ce manque d'exploration.

6.4.1 Détail de l'algorithme MOEGO NSGA-II

Nous proposons un algorithme qui combine les avantages des deux méthodes. Pour cela, l'idée proposée est de réaliser un algorithme séquentiel du type MOEGO en choisissant les points à rajouter au plan d'expériences à l'aide de l'algorithme NSGA-II, comme l'explique la figure 6.24. Plus précisément, à partir d'un plan d'expériences initial, par exemple de taille $N_{DOE}^0 = 10$, on construit un premier modèle de krigeage des fonctions objectif $\left(\hat{f}_i\right)_{1 \leqslant i \leqslant N_{obj}}$. L'algorithme NSGA-II est ensuite utilisé afin d'estimer le front de Pareto des approximations gaussiennes des fonctions objectif. Le temps d'exécution de cette étape est très réduit par rapport au temps d'exécution potentiel des objectifs. On peut donc laisser l'algorithme NSGA-II évoluer durant un nombre fixé de générations suffisamment grand. Par exemple, on peut prendre $N_{gen} = 500$ pour deux objectifs. Les points maximisant l'amélioration espérée sont ensuite recherchés sur la population afin d'enrichir le plan d'expériences. La section 6.4.2 explique le détail de cette étape. Ce nouveau plan d'expériences permet alors de construire un nouveau modèle de krigeage de chacune des fonctions objectif.

En ce qui concerne le critère EI multi-objectif utilisé, la section 6.3.2 montre que celui basé sur la distance euclidienne (voir équation (5.14)) donne de meilleurs résultats selon la mesure de convergence choisie dans cette étude. C'est donc celui qui est mis en place ici. Toutefois, la méthode proposée peut s'utiliser avec n'importe quelle version de l'EI multi-objectif. C'est d'ailleurs la raison pour laquelle les notations utilisées pour ce critère à la figure 6.24 restent générales. Enfin, la section précédente a montré que la modification de Parr du critère MOEI réduit la capacité d'exploration de la méthode MOEGO. De ce

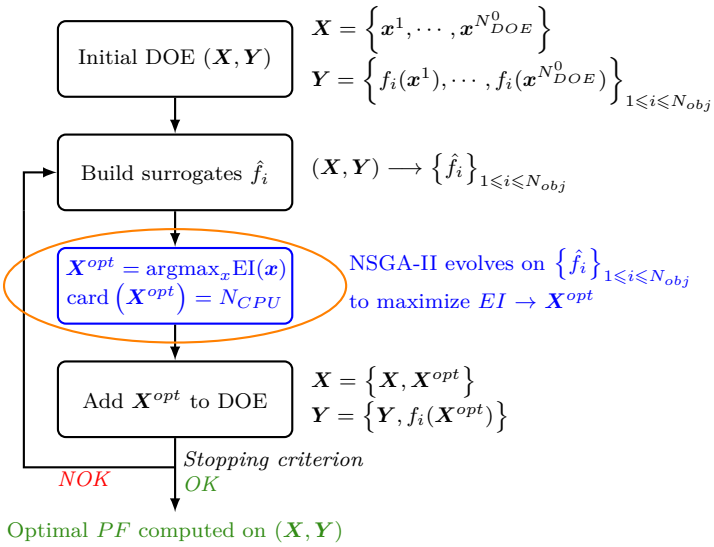


FIGURE 6.24 – Illustration de la méthode MOEGO NSGA-II

fait, le critère d'amélioration espérée par rapport au front complet, et non plus réduit à la zone dominée par un *goal point* comme introduit à la section 6.3.1, est utilisé.

Enfin, cet enrichissement du plan d'expériences aidé par NSGA-II est réalisé tant qu'un critère d'arrêt n'est pas atteint. Ce dernier correspond souvent à un budget de calcul alloué. Un autre critère d'arrêt peut aussi être défini en utilisant la valeur de l'amélioration espérée maximum à l'itération courante. En effet, on peut comparer ce maximum à une valeur caractéristique de même nature. Par exemple, dans le cas de la distance euclidienne, le maximum de l'amélioration espérée à l'itération courante peut être comparé à la distance minimale entre le front courant et le point de coordonnées $\mathcal{C} = \left(\min_{\boldsymbol{y} \in Y_{PF}} y_1, \min_{\boldsymbol{y} \in Y_{PF}} y_2\right)$. Dans le cas d'un critère basé sur l'hypervolume, le maximum de l'amélioration espérée peut être comparé à l'hypervolume du front courant. Ces deux distances sont représentées sur la figure 6.25.

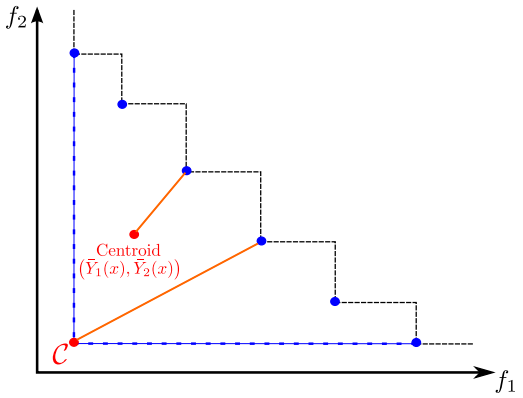


FIGURE 6.25 – Illustration des valeurs à comparer pour la construction d'un critère d'arrêt pour l'amélioration espérée basée sur la distance euclidienne

6.4.2 NSGA-II pour choisir les points d'enrichissement

Afin de distribuer les appels aux fonctions objectif, cette partie a pour but de détailler le choix simultané de plusieurs points pour l'enrichissement. Pour cela, on utilise NSGA-II afin d'explorer l'espace des entrées et d'approcher le PF sur le modèle de substitution entre chaque enrichissement. À partir de ce PF, l'enjeu est donc de choisir les points qui apportent l'information la plus pertinente, c'est-à-dire les points qui améliorent la connaissance du front de Pareto en divers endroits de l'espace des objectifs.

Une manière de choisir plusieurs points à chaque itération et permettant de forcer l'exploration est de choisir les points maximisant le critère EI répartis sur des sous-espaces $(\mathcal{A}_i)_{1\leqslant i\leqslant N_x}$ disjoints de l'espace des objectifs. Par exemple, avec deux objectifs, on peut partitionner l'espace des objectifs à l'aide de droites concentriques partant de l'origine. Ceci permet donc de répartir la population optimale en sous-ensembles de points. Voici les étapes du choix des points d'enrichissement utilisant NSGA-II :

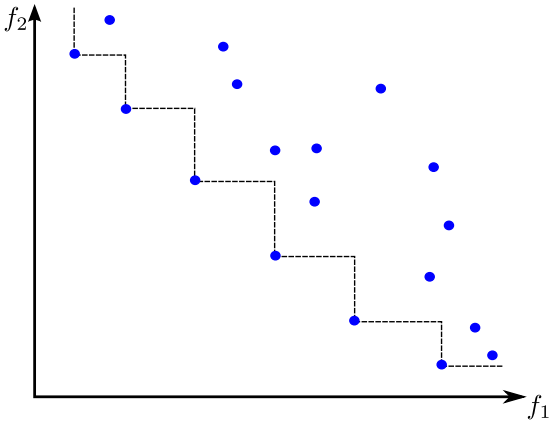


FIGURE 6.26 – Illustration de la population optimale $\hat{\mathcal{P}}$ obtenue sur les prédictions par krigeage des objectifs.

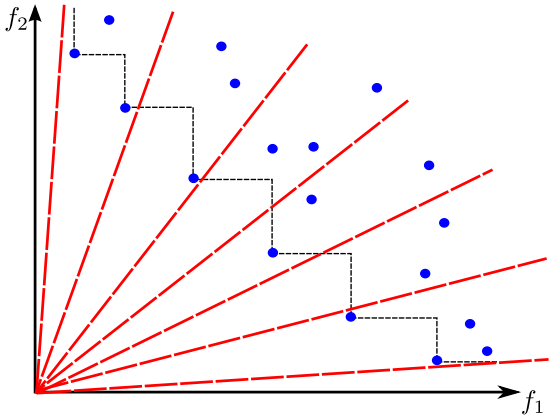


FIGURE 6.27 – Illustration de la décomposition de la population $\hat{\mathcal{P}}$ en sous-espaces disjoints $\hat{\mathcal{P}}_{\mathcal{A}_i}$ à l'aide de lignes concentriques

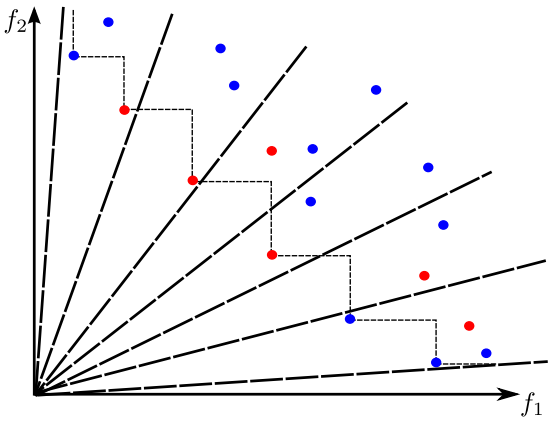


FIGURE 6.28 – Illustration du choix des points dans les sous-espaces disjoints $\hat{\mathcal{P}}_{\mathcal{A}_i}$

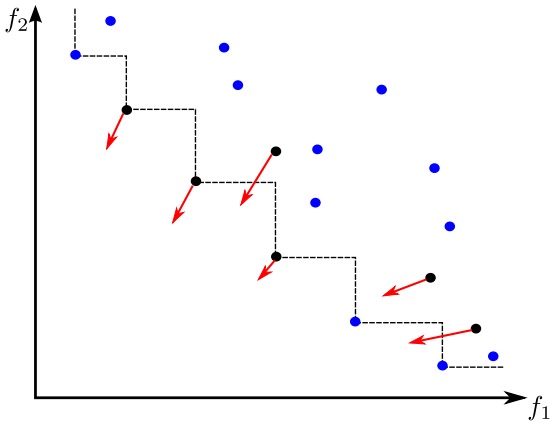


FIGURE 6.29 – Illustration de l'optimisation locale de l'EI à partir des points choisis à la figure 6.28

1. NSGA-II évolue pendant N_{gen} générations de sorte à minimiser $\hat{f}_1$ et $\hat{f}_2$ (la popu-

lation prédite $\hat{\mathcal{P}}$ après ces N_{gen} générations est représentée sur la figure 6.26).

$$\hat{\mathcal{P}} = \left\{ \left(\hat{f}_1(x_1), \hat{f}_2(x_1) \right), \cdots, \left(\hat{f}_1(x_{n_{POP}}), \hat{f}_2(x_{n_{POP}}) \right) \right\}$$

2. Sur cette population $\hat{\mathcal{P}}$, N_{CPU} points sont choisis, grâce aux N_{CPU} lignes convergentes représentées sur la figure 6.27. Ces points sont ensuite utilisés afin d'initialiser la recherche du maximum pour l'amélioration espérée. Ceci permet notamment de rajouter des points à divers endroits de l'espace des objectifs et obtenir un front bien distribué. Cette division de la population $\hat{\mathcal{P}}$ est discutée à la partie 6.4.5.1.

$$\hat{\mathcal{P}}_{\mathcal{A}_i} = \left\{ x_k \ : \ \left(\hat{f}_1(x_k), \hat{f}_2(x_k) \right) \in \mathcal{A}_i \right\}$$

3. On choisit ensuite les points maximisant l'amélioration espérée dans chacun des sous-espaces $\hat{\mathcal{P}}_{\mathcal{A}_i}$ (délimités par les lignes convergentes représentées sur la figure 6.28).

$$\begin{cases} \mathrm{EI}(\boldsymbol{x}) & = & \mathbb{E}\left[\mathrm{I}(\boldsymbol{x}) | Y(\boldsymbol{X}) = \boldsymbol{Y}\right] \\ \boldsymbol{x}_0^{k+1} & = & \mathrm{argmax}_{x \in \hat{\mathcal{P}}_{\mathcal{A}_1}} EI(x) \\ \boldsymbol{x}_0^{k+2} & = & \mathrm{argmax}_{x \in \hat{\mathcal{P}}_{\mathcal{A}_2}} EI(x) \\ & \cdots & \\ \boldsymbol{x}_0^{k+N_{CPU}} & = & \mathrm{argmax}_{x \in \hat{\mathcal{P}}_{\mathcal{A}_{N_{CPU}}}} EI(x) \end{cases}$$

4. Chacun de ces points est ensuite utilisé pour initialiser la recherche du maximum du critère EI à l'aide d'un optimiseur local (par exemple Nelder-Mead, détaillé en section 5.1.2.1) comme illustré sur la figure 6.29, et les optima obtenus sont ensuite rajoutés au plan d'expériences.

$$\begin{cases} \mathrm{EI}(\boldsymbol{x}) & = & \mathbb{E}\left[\mathrm{I}(\boldsymbol{x}) | Y(\boldsymbol{X}) = \boldsymbol{Y}\right] \\ \boldsymbol{x}^{k+1} & = & \mathrm{argmax}_{x \in V(\boldsymbol{x}_0^{k+1})} EI(x) \\ \boldsymbol{x}^{k+2} & = & \mathrm{argmax}_{x \in V(\boldsymbol{x}_0^{k+2})} EI(x) \\ & \cdots & \\ \boldsymbol{x}^{k+N_{CPU}} & = & \mathrm{argmax}_{x \in V(\boldsymbol{x}_0^{k+N_{CPU}})} EI(x) \\ \boldsymbol{X} & = & \left\{ \boldsymbol{X}, \boldsymbol{x}^{k+1}, \cdots, \boldsymbol{x}^{k+N_{CPU}} \right\} \\ \boldsymbol{Y}^i & = & \left\{ \boldsymbol{Y}^i, f_i(\boldsymbol{x}^{k+1}), \cdots, f_i(\boldsymbol{x}^{k+N_{CPU}}) \right\} \end{cases}$$

La notation $x \in V(\boldsymbol{x}_0^{k+i})$, $i \in \{1, \cdots, N_{CPU}\}$, pour le calcul de chacun de ces maxima permet uniquement de souligner le fait qu'un optimiseur local est utilisé, initialisé par le point $\boldsymbol{x}_0^{k+i}$. Toutefois, le domaine de recherche de ce dernier est $\mathcal{D}_x$ entier. Pour cette optimisation, puisque le critère d'amélioration espérée basé sur la distance euclidienne n'est pas dérivable, l'optimiseur COBYLA, développé par Powell [Powell, 1998], est utilisé.

6.4.3 Filtre des redondances à chaque itération

Puisque N_{CPU} points sont potentiellement ajoutés simultanément au plan d'expériences, certains de ces points peuvent donc être fortement corrélés, c'est-à-dire être proches dans l'espace des entrées et apporter une information redondante. En particulier, lors des

premières itérations de l'algorithme MOEGO NSGA-II, peu de points sont présents dans le DOE. À cause de cela, l'amélioration espérée peut avoir peu de maxima locaux et donc plusieurs des N_{CPU} maximisations locales peuvent retourner le même point. De ce fait, il est nécessaire de filtrer les points non informatifs avant de les inclure au plan d'expériences. L'estimation de l'erreur du krigeage $\hat{\sigma}$ peut ici être utilisée. En effet, elle ne dépend pas de la valeur de la sortie, comme expliqué en section 3.2.3.3. Un point peut être virtuellement ajouté au plan d'expériences du krigeage pour fournir l'estimateur de l'erreur $\hat{\sigma}^{+x}$ qu'aurait le modèle de krigeage une fois le point $\boldsymbol{x}$ rajouté. Le point est dit *virtuellement ajouté* puisque l'on n'utilise pas la valeur de la sortie en ce dernier. Ainsi, l'estimateur d'erreur virtuel $\hat{\sigma}^{+x}$ est utilisé sans recalculer les paramètres du modèle de krigeage (variance, longueurs de corrélation, etc...). Ceci est expliqué dans la section 3.2.4.1.

Pour utiliser cette erreur virtuelle du krigeage dans le but de filtrer les redondances parmi les N_{CPU} points, on part du dernier plan d'expériences DOE, représenté en figure 6.30. L'estimation de l'erreur « réelle » $\hat{\sigma}$ est représentée sur le graphe du bas. Les points sont ensuite testés séquentiellement : le point d'amélioration espérée maximale des N_{CPU} points est virtuellement ajouté (comme expliqué à la section 3.2.4.1). Sur le graphe du bas de la figure 6.31, l'erreur courante $\hat{\sigma}$ est représentée en pointillés alors que celle calculée par ajout virtuel $\hat{\sigma}^{+x}$ y est représentée en vert. On ajoute par la suite le point suivant (classé selon l'amélioration espérée), il est rejeté si sa variation d'écart type entre le plan d'expériences courant et l'écart type estimé sur le plan d'expériences virtuel est plus grande que $p_{reject} = 90\%$ (comme représenté sur la figure 6.32). Si sa variation est plus faible que p_{reject}, alors le point est gardé pour ajout, comme représenté sur la figure 6.33 puis 6.34.

Ce processus est répété tant que les N_{CPU} points n'ont pas été testés. Ceci est résumé à l'algorithme 6.1.

Algorithme 6.1 Algorithme de filtre des redondances des N_{CPU} points à l'aide de l'erreur virtuelle du krigeage.

Input : Le plan d'expériences courant DOE
Les entrées des N_{CPU} points à ajouter $\left(\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{N_{CPU}}\right)$
La valeur de l'amélioration espérée en ces points $\left(EI^1, \cdots, EI^{N_{CPU}}\right)$

Output : La liste $\boldsymbol{X}^{opt}$ des points à évaluer avec les fonctions objectif

· Classer les points à rajouter de manière décroissante par rapport à la valeur de l'EI : $\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(N_{CPU})}$;

· Rajouter le point maximisant l'EI à la liste des points à évaluer sur les objectifs $\boldsymbol{X}^{opt} = \left\{\boldsymbol{x}^{(1)}\right\}$;

for $2 \leqslant j \leqslant N_{CPU}$ **do**

  if $\forall i \in \{1, \cdots, N_{obj}\}$, $\frac{\hat{\sigma}^{+X^{opt}}(\boldsymbol{x}^{(j)})}{\hat{\sigma}(\boldsymbol{x}^{(j)})} > 0.1$ **then**

    · $\boldsymbol{X}^{opt} = \left\{\boldsymbol{X}^{opt}, \boldsymbol{x}^{(j)}\right\}$;

  end

end

· Retourner $\boldsymbol{X}^{opt}$;

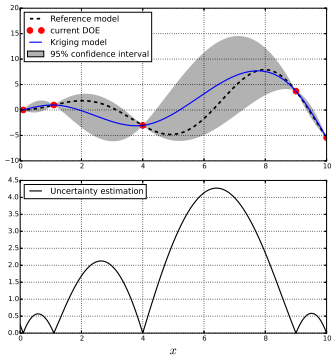


FIGURE 6.30 – État initial

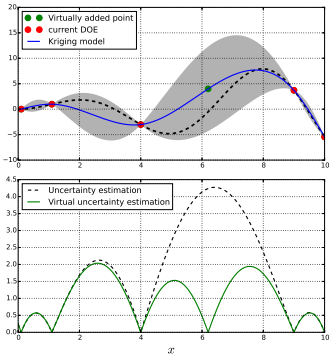


FIGURE 6.31 – Ajout virtuel du point d'amélioration espérée maximale.

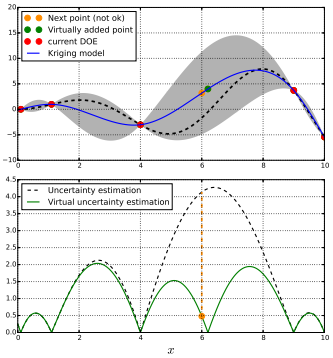


FIGURE 6.32 – Test du point suivant dont l'erreur a été grandement modifiée par l'ajout du premier point.

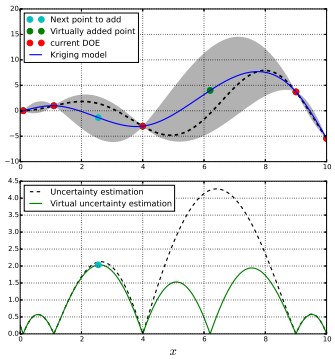


FIGURE 6.33 – Test du point suivant dont l'erreur a peu été modifiée par l'ajout du premier point.

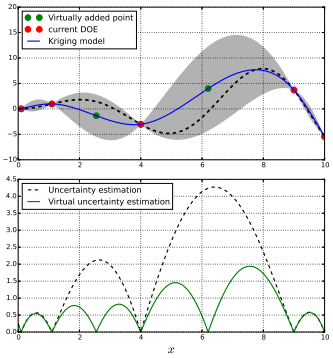


FIGURE 6.34 – Ajout virtuel du point accepté précédent.

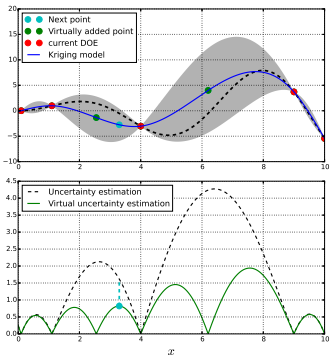


FIGURE 6.35 – Test du point suivant (classé décroissant selon l'amélioration espérée)

FIGURE 6.36 – Illustration du plan d'expériences initial et de l'effet de l'ajout virtuel de points au plan d'expériences sur l'estimateur de l'écart type de la prédiction fourni par le krigeage.

6.4.4 Résultats sur les cas analytiques

Les performances de l'algorithme NSGA-II sont maintenant testées sur les cas tests précédemment présentés. Pour cela, il faut choisir les différents paramètres de l'algorithme MOEGO NSGA-II, qui sont résumés sur la figure 6.37. On peut remarquer que le nombre de paramètres importants est relativement restreint. En effet, tout ce qui a trait à l'algorithme NSGA-II s'exécute sur une fonction peu coûteuse, il n'est donc pas nécessaire d'y porter trop d'efforts et il est facile de mettre des valeurs conservatives pour ces paramètres. Nous proposons par exemple une population de taille 100 et 500 générations. Ceci permet de s'assurer de la convergence avant l'ajout des points au plan d'expériences. Il ne reste donc que deux paramètres d'importance : la taille du plan d'expériences initial N^0_{DOE} et le

nombre de points N_{CPU} à ajouter au plan d'expériences à chaque itération de l'algorithme EGO. La taille du plan d'expériences initial est fixée à 10, ce qui est faible mais permet de garder une part importante du budget d'appels disponibles aux fonctions objectif afin de maximiser le critère d'amélioration espérée et donc d'enrichir les zones d'intérêt. Le nombre de processeurs correspond au nombre de points que l'on rajoute à chaque itération. Cette valeur ne peut pas être choisie trop grande au risque de perdre tout l'intérêt des algorithmes basés sur l'amélioration espérée. En effet, ce critère est optimal dans le cas de l'ajout d'un seul point séquentiellement. Or cet ajout nécessite théoriquement l'actualisation du modèle de krigeage. Plus N_{CPU} est grand, plus l'on s'éloigne de l'optimalité des points que l'on rajoute. Par ailleurs, ce paramètre N_{CPU} dépend également du budget de calcul disponible. Puisque l'on souhaite utiliser un outil de calcul de type ordinateur de bureau, on se limite à $N_{CPU} = 10$.

Pour que les résultats soient comparables à ceux de la méthode de Parr de la section précédente, le même plan d'expériences initial est utilisé pour les cinquante optimisations.

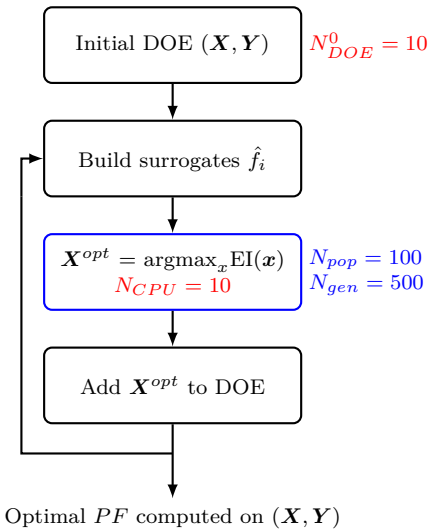


FIGURE 6.37 – Résumé des paramètres de l'algorithme NSGA-II et leur valeur pour l'application sur les cas analytiques

Comme précédemment, les résultats sont donnés par l'intermédiaire des courbes de la distance au vrai front en fonction du nombre d'appels distribués aux objectifs. Ils sont donnés sur les figures 6.38 à 6.41. Ces figures comparent la méthode MOEGO avec la formule de Parr de la section précédente et l'assistance par NSGA-II présentée dans cette partie. Dans les deux cas, $N_{CPU} = 10$. La première remarque que l'on peut faire est que l'utilisation de NSGA-II afin de choisir les points à ajouter au plan d'expériences permet en moyenne de rendre la méthode plus parcimonieuse pour atteindre un front de qualité satisfaisante.

En ce qui concerne la montée en dimension, elle semble être mieux amortie, notamment dans le cas de ZDT3 où l'on remarque que la méthode nécessite peu d'appels pour atteindre la distance cible. Par ailleurs, pour mettre en évidence que l'assistance par NSGA-II et l'utilisation de lignes concentriques permettent d'améliorer l'exploration de l'espace des objectifs, la figure 6.42 montre le résultat de l'une des optimisations dans le cas de ZDT3

MOP2 en dimension 4 :

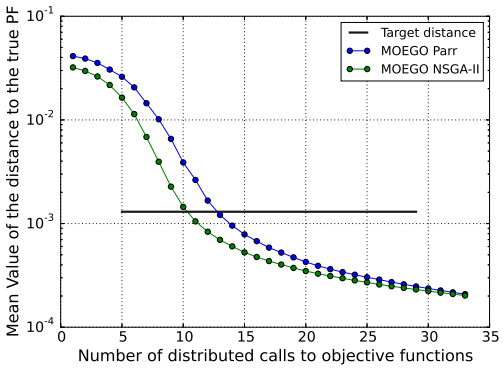


FIGURE 6.38 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 4

MOP2 en dimension 10 :

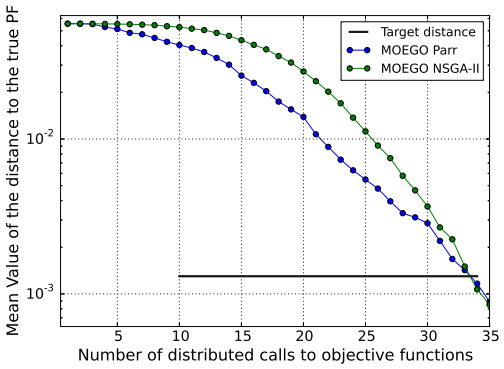


FIGURE 6.39 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 10

ZDT3 en dimension 4 :

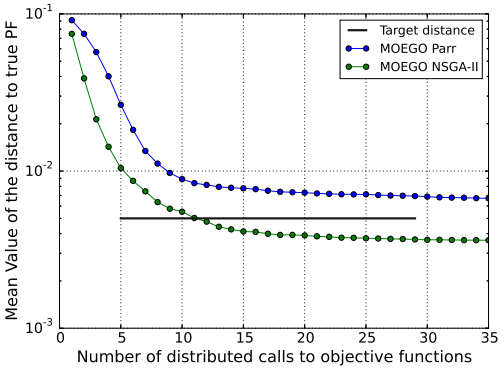


FIGURE 6.40 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 4

ZDT3 en dimension 10 :

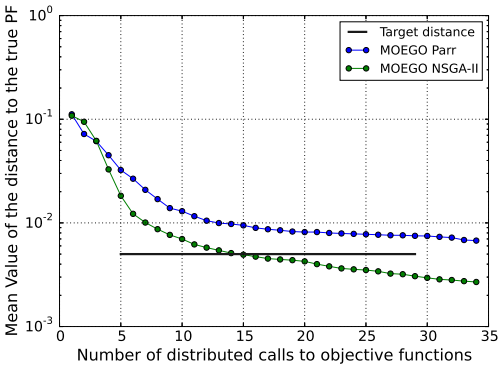


FIGURE 6.41 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 10

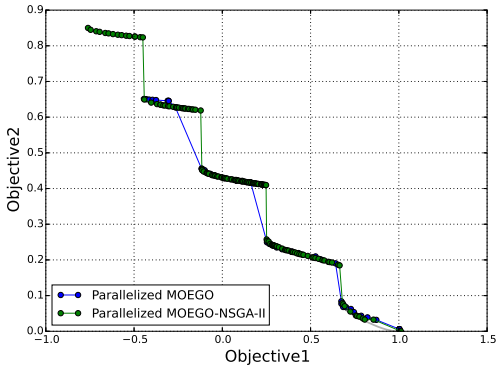


FIGURE 6.42 – Illustration d'un PF obtenu après 35 appels distribués pour le cas test ZDT3 en dimension 10

	NSGA-II	MOEGO NSGA-II
MOP2 4D	130	15
MOP2 10D	735	40
ZDT3 4D	150	15
ZDT3 10D	385	35

TABLEAU 6.3 – Tableau récapitulant le nombre d'appels nécessaires afin que la valeur maximum sur 50 optimisations de la distance atteigne la distance cible

en dimension 10. On y remarque que le front entier est capturé alors que la méthode de Parr ne permet pas de l'obtenir et n'en capture qu'une petite partie (comme discuté dans la section précédente).

Concernant la robustesse de la méthode, nous comparons MOEGO NSGA-II avec la méthode NSGA-II dont nous avons déjà prouvé la robustesse. Pour cela, les figures 6.43 à 6.46 permettent de comparer les boîtes à moustaches de la distance au front des cinquante optimisations après un certain nombre d'appels distribués. Ceci permet de résumer le nombre d'appels distribués nécessaires à la convergence entre NSGA-II et MOEGO NSGA-II dans le tableau 6.3.

On remarque que l'utilisation du critère d'amélioration espérée maximisé à l'aide de NSGA-II converge, en peu d'appels aux fonctions objectif, vers un PF de qualité et bien distribué comme le montre la figure 6.42.

Enfin, afin de montrer que le filtre de redondances est utile, les courbes des figures 6.47 et 6.48 illustrent le nombre **moyen** de points ajoutés au plan d'expériences à chaque appel distribué. On remarque que pour les premières itérations de l'algorithme NSGA-II, il est rare que les N_{CPU} points choisis soient tous ajoutés au plan d'expériences. Ceci s'explique par le fait que les points du front courant sont initialement mal placés dans l'espace des objectifs et peuvent être très proches les uns des autres. Les lignes concentriques sont définies par rapport aux points extrêmes du front courant dans l'espace des objectifs et donc beaucoup des N_{CPU} points des premières itérations sont corrélés. La différence entre ZDT3 et MOP2 donne justement raison à cela puisque dans le cas où la fonction MOP2 est faiblement échantillonnée, la probabilité est grande que les points optimaux se trouvent

MOP2 en dimension 4 :

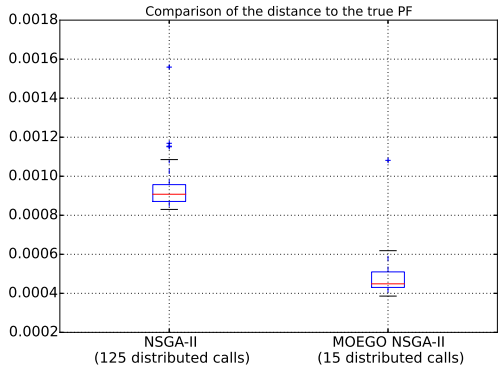


FIGURE 6.43 – Graphique représentant les boîtes à moustaches de la distance pour le cas test MOP2 en dimension 4

MOP2 en dimension 10 :

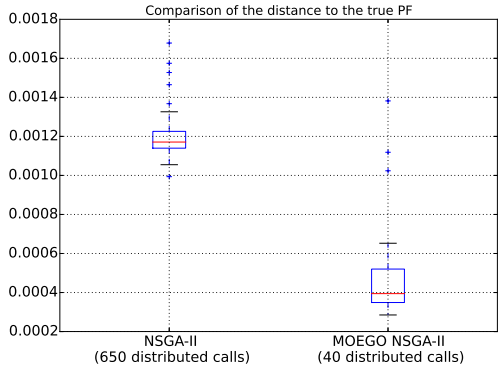


FIGURE 6.44 – Graphique représentant les boîtes à moustaches de la distance pour le cas test MOP2 en dimension 10

ZDT3 en dimension 4 :

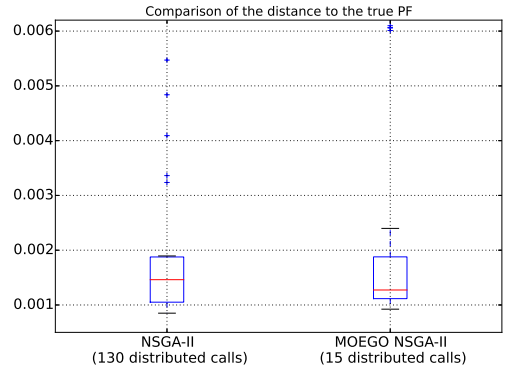


FIGURE 6.45 – Graphique représentant les boîtes à moustaches de la distance pour le cas test ZDT3 en dimension 4

ZDT3 en dimension 10 :

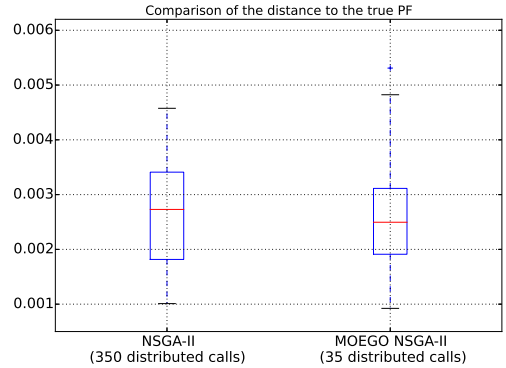


FIGURE 6.46 – Graphique représentant les boîtes à moustaches de la distance pour le cas test ZDT3 en dimension 10

à proximité du point $(1, 1)$ dans l'espace des objectifs. Ceci est d'autant plus vrai que la dimension des entrées est grande. De ce fait, le découpage de l'espace des objectifs en sous-ensembles se fait sur une zone très réduite de l'espace des objectifs et les optima locaux trouvés sont ainsi très proches les uns des autres. De ces figures, on peut donc en conclure que l'algorithme NSGA-II permet en peu d'itérations de proposer des points d'initialisation pour le calcul du maximum de l'EI suffisamment diversifiés. C'est ce qui explique la convergence rapide de MOEGO NSGA-II.

MOP2 :

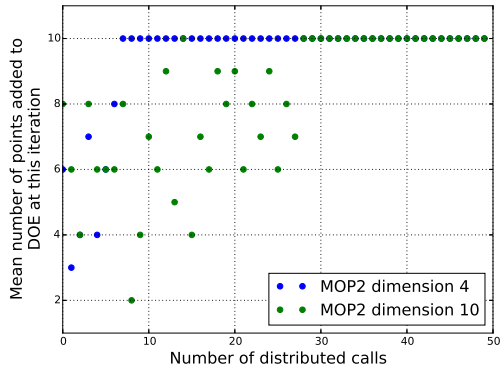


FIGURE 6.47 – Nombre moyen de points ajoutés au plan d'expériences après le nombre d'appels distribués donné en abscisse.

ZDT3 :

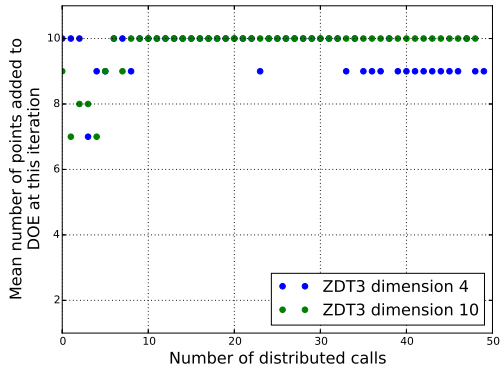


FIGURE 6.48 – Nombre moyen de points ajoutés au plan d'expériences après le nombre d'appels distribués donné en abscisse.

6.4.4.1 Comparaison entre le point de vue MOEGO NSGA-II ou NSGA-II assisté par krigeage

La section 5.2.2 présente deux manières d'utiliser un modèle de substitution en optimisation multi-objectif. La première, basée sur l'amélioration espérée, consiste à diriger l'optimisation à l'aide du modèle de substitution. De ce fait, le PF retourné est celui calculé sur les points du plan d'expériences. C'est le point de vue adopté ici, c'est la raison pour laquelle il s'agit d'une méthode de type MOEGO assisté par NSGA-II.

La seconde consiste à utiliser le modèle de substitution comme un pur substitut des objectifs dans le cadre d'un algorithme évolutionnaire. Avec cette vision, l'enrichissement du plan d'expériences correspond à une mise à jour des modèles de substitution. Ceci signifie que la population obtenue sur le modèle de substitution lors de la dernière génération est le PF retourné par l'algorithme. On peut éventuellement évaluer ces points sur les vraies fonctions objectif. Cette évaluation peut s'avérer indispensable si la variance du krigeage des fonctions objectif est élevée sur les individus de cette population.

L'algorithme présenté ici peut donc potentiellement être vu comme un algorithme NSGA-II assisté par krigeage. L'étape d'enrichissement détaillée à la section 6.4.2 peut en effet être interprétée comme un mise à jour des modèles de krigeage des fonctions objectif. Toutefois, un tel algorithme a pour inconvénient de retourner des points estimés optimaux sur la surface de réponse. Ceci implique que ces points peuvent nécessiter d'être estimés sur la vraie fonction alors que l'approche EGO ne retourne que des points estimés sur les vraies fonctions objectif.

FIGURE 6.49 – Comparaison du PF selon que l'on retourne les points Pareto-optimaux du plan d'expériences (MOEGO NSGA-II) ou que l'on retourne les points de la population à la dernière génération de NSGA-II (NSGA-II assisté par krigeage).

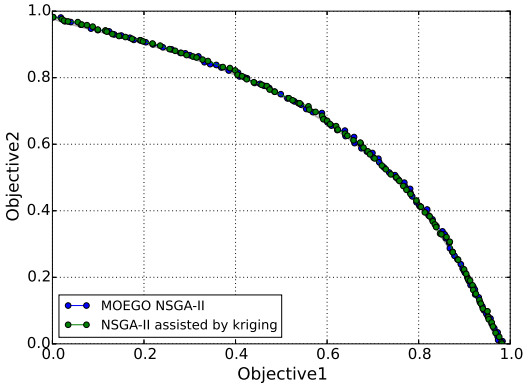


FIGURE 6.50 – MOP2 en dimension 10.

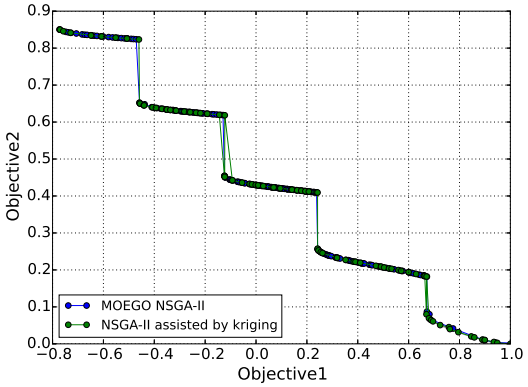


FIGURE 6.51 – ZDT3 en dimension 10.

Sur nos simulations, les deux formulations semblent donner des résultats comparables. Les figures 6.50 et 6.51 montrent que dans les deux visions, le front de Pareto semble bien discrétisé. Toutefois, la vision NSGA-II assisté par krigeage retourne un PF de taille fixée, restreint à la taille de la population à la manière d'un algorithme génétique classique. Ces points étant évalués sur les prédictions par krigeage des fonctions objectif, il peut être nécessaire de les recalculer, ce qui constitue un inconvénient. À l'opposé, la version MOEGO assistée par NSGA-II ne retourne que les points Pareto-optimaux du plan d'expériences, ce qui signifie qu'ils ont été estimés sur les fonctions de référence. Il n'y a donc pas besoin de refaire appel aux fonctions objectif coûteuses.

Dans tous les cas, c'est un autre avantage de la méthode que de fournir deux manières de calculer le front de Pareto. Par exemple, si l'une des deux ne donne pas une discrétisation satisfaisante, on peut la compléter par l'autre. Ceci est notamment applicable pour ZDT3, sur la figure 6.51. En effet, on peut remarquer sur ce graphique que dans la zone du front où $0.65 \leqslant f_1 \leqslant 1.0$, la version NSGA-II assistée par krigeage permet de rajouter des points Pareto-optimaux qui n'avaient pas été rajoutés au plan d'expériences, notamment parce que l'estimation d'erreur du krigeage en ces points était très faible (de l'ordre de 10^{-7}). Or, la partie où $-0.8 \leqslant f_1 \leqslant 0.65$ est beaucoup mieux discrétisée dans le cas de la méthode MOEGO.

On peut donc enrichir le PF retourné par la méthode MOEGO NSGA-II à l'aide de la population NSGA-II prédite par le krigeage à la fin de l'algorithme. Si des points de cette dernière se trouvent dans des zones du PF où la discrétisation est mauvaise, il peut être intéressant d'y rajouter ces points. Dans le cas où les estimateurs d'erreur $(\hat{\sigma}_i)_{1 \leqslant i \leqslant N_{obj}}$ de la prédiction des objectifs en ces nouveaux individus sont faibles (en dessous de 10^{-5} par exemple), nous pouvons même envisager de les retourner sans les évaluer sur les fonctions de référence.

Enfin, il y a un autre cas dans lequel la version NSGA-II assistée par krigeage peut être utile : lorsque la fonction est bruitée. Dans ce cas, le modèle de krigeage peut être utilisé avec un effet pépite le rendant non interpolant. Même si l'amélioration espérée est

encore utilisée pour enrichir le plan d'expériences, il est dès lors préférable de retourner la valeur prédite par le krigeage prenant en compte le bruit plutôt que la valeur exacte provenant du code (qui est bruitée).

6.4.4.2 Test de prolongement de la méthode

L'application de l'algorithme NSGA-II sur les prédictions par krigeage $\left(\hat{f}_i\right)_{1\leqslant i\leqslant N_{obj}}$ sert donc à explorer intensivement l'espace des objectifs. De sorte à intensifier l'exploration, l'idée a été proposée d'utiliser une formule plus optimiste et d'appliquer NSGA-II sur la borne de confiance inférieure de la prédiction par krigeage $\left(\hat{f}_i - k\hat{\sigma}_i\right)_{1\leqslant i\leqslant N_{obj}}$. Le coefficient k permet de définir le pourcentage de l'intervalle de confiance utilisé. Afin de quantifier l'influence de ce coefficient, deux méthodes ont été testées. Pour la première, ce coefficient k est fixé à 1.96, ce qui permet de minimiser la borne de confiance inférieure à 95%. Pour la seconde, nous avons choisi de faire décroître la valeur de k avec le nombre de points dans le plan d'expériences, pour atteindre une valeur nulle à l'approche du budget total d'appels. Comme dans la première procédure, la valeur initiale de k est fixée à 1.96. Les résultats en dimension 10 sont donnés sur les figures 6.52 et 6.53. Ces résultats mettent en évidence le manque de robustesse de cette technique. Quelle que soit la stratégie utilisée pour le choix de k, la figure 6.52 illustre le fait que l'introduction de l'écart type peut drastiquement ralentir la convergence alors qu'elle l'accélère peu dans les cas favorables tels que ZDT3 sur la figure 6.53.

MOP2 :

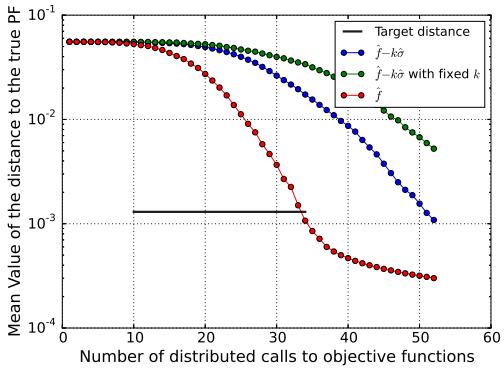


FIGURE 6.52 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test MOP2 en dimension 4

ZDT3 :

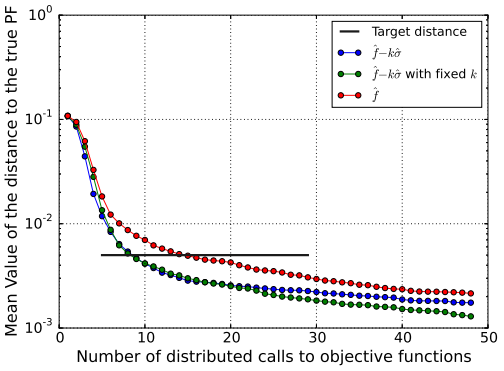


FIGURE 6.53 – Graphique représentant la distance entre $PF^\star$ et PF en fonction du nombre d'appels aux fonctions objectif pour le cas test ZDT3 en dimension 4

La raison de la non-robustesse de cette méthode provient du fait qu'elle enrichit des zones d'incertitudes grandes mais qui n'améliorent pas forcément la connaissance du front. Les cas favorables sont donc ceux où les zones de fortes incertitudes, telles que les bords du domaine, correspondent à des points critiques de la fonction objectif. Or, ce n'est pas le cas de MOP2 où les zones critiques se situent au milieu du domaine. De ce fait, les points vers lesquels mènent la population $\hat{\mathcal{P}}$ ne permettent pas systématiquement d'améliorer l'approximation de $PF^\star$.

Pour résumer, du fait de la non robustesse de cette solution et sans *a priori* sur les fonctions à minimiser, l'utilisation de $\left(\hat{f}_i - k\hat{\sigma}_i\right)_{1\leqslant i\leqslant N_{obj}}$ à la place de $\left(\hat{f}_i\right)_{1\leqslant i\leqslant N_{obj}}$ dans la partie d'enrichissement du plan d'expériences par NSGA-II est fortement déconseillée.

6.4.5 Conclusions sur la méthode MOEGO NSGA-II

Pour résumer, voici les apports de la méthode MOEGO NSGA-II pour l'optimisation par modèles de substitution et calcul distribué :

1. Elle nécessite peu d'appels distribués aux objectifs afin de converger vers un PF de qualité suffisante.
2. Elle permet de distribuer les appels aux objectifs dans le cas où le nombre de processeurs disponibles est réduit. Ce chiffre dépend de la dimension mais il est préférable de rajouter moins de 10 points à chaque itération.

Enfin, dans le cas où le résultat de la méthode MOEGO n'est pas satisfaisant, il est possible de l'enrichir à l'aide de la population NSGA-II optimale à la fin de l'algorithme. Cette population étant prédite par krigeage. Il est alors important de réévaluer ces points s'ils ont une variance de krigeage trop importante. L'utilisation de cette population NSGA-II finale peut notamment être utile si au moins l'une des fonctions objectif est bruitée.

6.4.5.1 Extension du domaine d'application de MOEGO NSGA-II

Cette méthode peut être adaptée à la présence de contraintes ainsi qu'au cas où le nombre d'objectifs est supérieur à deux. Toutefois, cette extension nécessite quelques adaptations de l'algorithme MOEGO NSGA-II.

Dans le cas de plus de deux objectifs : il y a deux paramètres à modifier dans l'algorithme détaillé précédemment.

1. La manière de calculer le critère d'amélioration espérée. En effet, au-delà de deux voire trois objectifs, il faut estimer le critère à l'aide de méthodes de Monte-Carlo. C'est notamment ce qui est fait dans l'étude de Féliot [Féliot *et al.*, 2016]. Pour cela, des espérances sont à estimer. La taille de l'échantillon Monte-Carlo n'est donc pas nécessairement très élevée. On peut par exemple utiliser l'estimateur de la variance de l'erreur commise par cet estimateur Monte-Carlo afin de déterminer si la précision est suffisante.
2. Le partitionnement de la population optimale de NSGA-II par des ensembles disjoints $(\mathcal{A}_i)_{1\leqslant i\leqslant N_{CPU}}$. Dans le cas de deux objectifs, il est aisé de construire des lignes concentriques de sorte à choisir les points dans des zones disjointes de l'espace. En dimension plus élevée, on peut encore découper l'espace des fonctions objectif à l'aide d'hyperplans séparateurs (l'équivalent en dimension supérieures des droites) concentriques. Une autre idée est d'utiliser un *clustering* afin de séparer les N_{pop} points de la population prédite optimale sur NSGA-II, $\hat{\mathcal{P}}$. De ce fait, il suffit de choisir N_{CPU} *clusters* pour scinder la population en sous-populations disjointes $\hat{\mathcal{P}}_{A_i}$. Ensuite, l'étape de maximisation de l'amélioration espérée reste la même que précédemment.

Pour la prise en compte des contraintes : Un problème multi-objectif sous contraintes s'écrit de la manière suivante :

$$\begin{cases} \min\limits_{x} f_1(\boldsymbol{x}) \\ \min\limits_{x} f_2(\boldsymbol{x}) \\ \text{s.c } \boldsymbol{G}(\boldsymbol{x}) \leqslant 0 \end{cases} \tag{6.10}$$

$\boldsymbol{G}(\boldsymbol{x}) = (g_1(\boldsymbol{x}), \cdots, g_{N_{cons}}(\boldsymbol{x}))$ sont les fonctions contraintes. Les points vérifiant les contraintes sont dits points admissibles, les points ne les vérifiant pas sont dits infaisables.

MOEGO NSGA-II peut prendre en compte les contraintes, même dans le cas où les points initiaux sont infaisables. Pour cela, il est nécessaire de pénaliser les fonctions objectif ou de modifier la définition de la dominance dans le cas de NSGA-II et d'adapter la formule du calcul de l'amélioration espérée. Pour NSGA-II, on peut par exemple citer l'étude de Deb [Deb, 2000]. Il propose de pénaliser fortement les fonctions objectif lorsque la solution est infaisable. Pour cela, il faut que cette pénalisation respecte trois règles :

1. si l'on compare deux solutions faisables, alors on utilise la dominance de Pareto classique,
2. si l'on compare une solution faisable et une solution infaisable, la solution faisable doit toujours être privilégiée,
3. si l'on compare deux solutions infaisables, il faut privilégier la solution violant le moins les contraintes.

Le point qui nécessite le plus d'aménagement de la dominance classique est le point 3. En effet, il requiert d'être capable d'ordonner les contraintes. Ce que propose Deb dans son étude [Deb, 2000] est donc de pénaliser la fonction coût de la manière suivante :

$$\boldsymbol{F}^{pen}(\boldsymbol{x}) = \begin{cases} \boldsymbol{F}(\boldsymbol{x}) & \text{si } \boldsymbol{G}(\boldsymbol{x}) \leqslant 0 \\ \boldsymbol{F}^{max} + \sum_{j=1}^{N_{cons}} \max\left(g_j(\boldsymbol{x}), 0\right) & \text{si } \exists 1 \leqslant j \leqslant N_{cons} \ : \ g_j(\boldsymbol{x}) \geqslant 0 \end{cases} \tag{6.11}$$

L'inconvénient de cette technique est qu'elle pénalise les points infaisables par une formulation scalaire. Plus précisément, elle résume en une seule équation toutes les violations de contraintes. Or, ceci nécessite que les contraintes soient comparables et on peut préférer la méthode choisie par Féliot [Féliot *et al.*, 2016] dans son étude. Elle consiste à remplacer la dominance de Pareto $\succ$ par la dominance notée $\rhd$. Celle-ci est définie de la manière suivante : $(\boldsymbol{y}, \boldsymbol{y}^c) \rhd (\boldsymbol{y}', \boldsymbol{y}^{c\prime})$ si $\Gamma(\boldsymbol{y}, \boldsymbol{y}^c) \succ \Gamma(\boldsymbol{y}', \boldsymbol{y}^{c\prime})$. La fonction Γ est définie de la manière suivante :

$$\begin{array}{rcccl} \Gamma & : & \mathbb{R}^{N_{obj}} \times \mathbb{R}^{N_{cons}} & \longrightarrow & \mathbb{R}^{N_{obj}} \times \mathbb{R}^{N_{cons}} \\ & & (\boldsymbol{y}, \boldsymbol{y}^c) & \longmapsto & \begin{cases} (\boldsymbol{y}, 0) & \text{si } \boldsymbol{y}^c \leqslant 0 \\ (+\infty, \max(\boldsymbol{y}^c, 0)) & \text{sinon} \end{cases} \end{array} \tag{6.12}$$

Cette nouvelle dominance vérifie les trois points présentés précédemment. En effet, si l'on compare deux points admissibles, cela revient à utiliser la dominance de Pareto. Si l'on compare un point admissible et un point non admissible, le point admissible domine forcément l'autre. Enfin, si l'on compare deux points non-admissibles, on utilise la dominance de Pareto dans l'espace des contraintes violées.

Pour l'amélioration espérée, il existe plusieurs façons de modifier la formule de sorte à prendre en compte les contraintes. Notamment, la redéfinition de la dominance précédente par Féliot est utilisée pour construire un nouveau critère d'amélioration espérée.

Toutefois, puisque l'amplitude de la violation de la contrainte est déjà mesurée par l'évolution NSGA-II, il est possible d'utiliser une version plus simple de l'EI, en multipliant le critère sans contrainte par la probabilité que les contraintes soient respectées. Soient $\boldsymbol{Y}^c(\boldsymbol{x}) = \left(Y_1^c(\boldsymbol{x}), \cdots, Y_{N_{cons}}^c(\boldsymbol{x})\right)$ les processus gaussiens définis par le krigeage (définis tels qu'à l'équation (3.16)) des fonctions contraintes. Si l'on fait l'hypothèse que les contraintes sont indépendantes entre elles et avec les fonctions objectif, le nouveau critère d'amélioration espérée peut s'écrire :

$$\mathcal{I}\text{nfo}(\boldsymbol{x}) = \mathbb{E}(I_\bullet)\,\mathbb{P}(\boldsymbol{Y}^c(\boldsymbol{x}) \leqslant 0) \tag{6.13}$$

La notation $I_\bullet$ pour l'amélioration traduit le fait que cette manière de prendre en compte la contrainte fonctionne aussi bien pour l'amélioration basée sur l'hypervolume détaillée à l'équation (5.16), que pour celle basée sur la distance euclidienne détaillée à l'équation (5.14). Cette idée est notamment proposée dans l'étude de Couckuyt [Couckuyt *et al.*, 2014]. Ce critère présente l'avantage de rajouter un terme calculable analytiquement puisque $\boldsymbol{Y}^c$ est un processus gaussien de loi connue analytiquement.

Pour résumer, afin de prendre en compte les contraintes il faut :

- Faire un modèle de krigeage des contraintes.
- Dans NSGA-II :
  - ◇ Pénaliser les fonctions objectif tel qu'à l'équation (6.11) ou modifier la dominance de Pareto tel qu'à l'équation (6.12).
  - ◇ Répartir les points de la population optimale par *clustering* (à l'aide de N_{CPU} *clusters*).
- Pour l'amélioration espérée : modifier la formule du calcul de l'amélioration espérée, tel qu'à l'équation (6.13).

6.4.5.2 Limites de l'algorithme MOEGO NSGA-II

La plupart des limites de cet algorithme sont héritées de la construction de modèles de substitution par krigeage. De ce fait, la principale restriction concerne le nombre de paramètres d'entrées. En effet, le krigeage a une efficacité garantie en dessous de la dimension 10. Au-delà, plusieurs facteurs peuvent en dégrader les performances :

- Tout d'abord, un problème qui est commun à toutes les techniques de réduction de modèle est le fléau de la dimension. Il est par exemple illustré dans le livre [Hastie *et al.*, 2009]. En effet, à partir de la dimension 10, les points sont tous à des distances importantes les uns des autres, en conduisant à des points « isolés ». Pour compenser cela, il est de ce fait nécessaire d'utiliser des plans d'expériences de taille plus importante. Or, l'augmentation avec la dimension de la distance entre les points n'est pas linéaire et la taille du plan d'expériences nécessaire pour capter certains effets peut devenir irréalisable. Dans le cas du krigeage, le corollaire de cette augmentation de distance entre les points implique que l'estimateur d'erreur du krigeage est plus élevé en moyenne avec la dimension. De plus, cet estimateur d'erreur a tendance à moins diminuer avec l'ajout de points au plan d'expériences.
- L'augmentation en dimension implique donc que la taille du plan d'expériences doit être plus importante. Par ailleurs, la taille du plan d'expériences détermine la taille de la matrice de covariance du krigeage. Or, cette matrice doit être inversée pour estimer les paramètres du modèle. De plus, la prédiction par krigeage nécessite également l'utilisation de cette matrice. De ce fait, plus la dimension de $\boldsymbol{x}$ est

importante, plus la construction et l'exécution du modèle de krigeage est lente. Cette lenteur dans l'exécution pose problème dans l'algorithme MOEGO puisque le calcul et la maximisation de l'amélioration espérée nécessitent beaucoup d'appels à ce modèle. Dès lors, ceci peut rendre la méthode inefficiente.

Par ailleurs si le nombre de processeurs disponible est grand, les calculs de la section 6.2 montrent que des algorithmes évolutionnaires peuvent être compétitifs. En effet, ce type d'algorithme permet de paralléliser de nombreux calculs dans le cas où le nombre de processeurs accessibles n'est pas limité. Par exemple, dans le cas de la dimension 4 à la section 6.2, si 50 processeurs étaient disponibles, la résolution du problème n'aurait nécessité qu'une trentaine d'appels distribués.

En outre, la méthode MOEGO n'est pas adaptée à la distribution d'un trop grand nombre de calculs : l'ajout d'un trop grand nombre de points simultanément ne se justifie plus théoriquement puisque il est important de mettre à jour régulièrement le modèle de krigeage. Or, ceci n'est pas possible si l'on souhaite distribuer un nombre important d'appels.

Pour résumer, il existe deux cas dans lesquels la méthode proposée ici n'est pas conseillée :

1. Si le nombre de dimensions est trop important. La limite couramment donnée, provenant du fléau de la dimension, est 10.
2. Si le nombre de processeurs disponibles est très important (ce nombre dépendant de la dimension), la méthode proposée ici peut ne plus diminuer l'efficacité relative de MOEGO NSGA-II par rapport à des méthodes ne faisant pas appel à des modèles de substitution.

Chapitre 7

Optimisation Multi-objectif Pire cas

Sommaire

Ce chapitre a pour but de proposer un algorithme de résolution de l'équation (7.1).

$$\begin{cases} \min\limits_{x\in\mathcal{D}_x} \max\limits_{\xi\in\mathcal{D}_\xi} [f_1(\boldsymbol{x},\boldsymbol{\xi}_x,\boldsymbol{\xi}_e)] \\ \min\limits_{x} f_2(\boldsymbol{x}) \end{cases} \tag{7.1}$$

Nous nous plaçons dans le cas où $\boldsymbol{x} \in \mathcal{D}_x \subset \mathbb{R}^4$ et où $\boldsymbol{\xi} \in \mathcal{D}_\xi \subset \mathbb{R}^8$. Un tel algorithme permet ensuite de répondre au problème présenté à l'équation (2.12). La raison pour laquelle le pire cas est une mesure de robustesse intéressante pour les ingénieurs est qu'elle permet d'assurer que le système fonctionne quelles que soient les valeurs prises par les incertitudes sur les paramètres étudiés. Cet algorithme doit être applicable même dans le cas où les fonctions coût ont des temps d'exécution importants. De ce fait, pour diminuer le temps de restitution de la méthode, il est nécessaire qu'elle permette de distribuer les appels aux deux fonctions objectif. De plus, le nombre d'appels distribués nécessaires à ces fonctions doit également être minimisé. La section 7.1 permet de répondre à cela.

Nous appliquons ensuite l'algorithme proposé sur le cas industriel en section 7.2.

7.1 Résolution d'un problème multi-objectif pire cas

La section 5.3.2 liste toutes les techniques permettant de résoudre des problèmes mono-objectifs ou multi-objectifs avec des mesures de robustesse de type pire cas pour les incertitudes. Toutefois, aucune de ces méthodes n'est applicable en l'état. Par exemple, la relaxation proposée par Marzat [Marzat *et al.*, 2012], ou l'utilisation de modèles de substitution proposée par Rehman [Rehman *et al.*, 2014] ne sont conçues que pour la résolution de problèmes mono-objectifs. En ce qui concerne les techniques de type coévolutionnaire (voir la section 5.3.2.2), elles ont un coût important. Il serait donc nécessaire de mettre en place des techniques basées sur les modèles de substitution. Pour cela, le modèle de substitution étant une approximation, il faut s'assurer de la qualité de ce dernier en cours d'optimisation et ce sur les deux populations évoluant en parallèle. Cette mise en place peut être lourde à gérer. De plus, la méthode est *a priori* développée pour le cas où toutes les fonctions objectif sont incertaines et traitées par pire cas. Ceci nécessite donc une modification dans l'utilisation de la population de test. Dernière limite de cette méthode, elle n'est applicable que dans le cas où les incertitudes sont traitées de manière déterministe, donc par l'intermédiaire d'un pire cas. Le passage à une mesure de risque probabiliste n'est pas envisageable puisque l'utilisation d'une population de test, comme en coévolutionnaire, n'est pas adaptée à la prise en compte de la densité de probabilité des incertitudes en entrée.

Nous devons donc proposer de nouvelles techniques, en se basant sur ce que nous avons déjà développé. En particulier, l'algorithme multi-objectif MOEGO NSGA-II présenté en section 6.4 permet de retourner un front de Pareto satisfaisant en peu d'appels aux fonctions objectif. Néanmoins, cette technique étant basée sur l'utilisation de modèles de substitution, dont la forme analytique est continue, il est préférable de s'assurer de la continuité des fonctions objectif. La section 7.1.1 montre donc la continuité de l'application pire cas. Ensuite, il reste donc à choisir l'algorithme d'optimisation mono-objectif le plus adapté à l'estimation du pire cas en chaque point de contrôle $\boldsymbol{x}$ (section 7.1.2) et à coupler cette estimation de pire cas avec MOEGO NSGA-II (réalisé en section 7.1.3). Pour cela, le cadre du calcul de ce maximum est explicité et permet de choisir l'optimiseur mono-objectif le plus adapté à la problématique. Enfin, l'algorithme est détaillé et validé sur un cas analytique.

7.1.1 Preuve de la continuité de l'application pire cas

Notre premier théorème prouve la continuité du pire cas en faisant l'hypothèse que le domaine de définition de $\boldsymbol{\xi}$ ne dépend pas de $\boldsymbol{x}$:

Théorème 7.1. *Si $f \in \mathcal{C}^0\left(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R}\right)$ avec $\mathcal{D}_x \subset \mathbb{R}^{N_x}$ et $\mathcal{D}_\xi \subset \mathbb{R}^{N_\xi}$ deux ensembles compacts. Alors la fonction g définie à l'équation (7.2) est continue pour tout $\boldsymbol{x} \in \mathcal{D}_x$:*

$$g(\boldsymbol{x}) = \max_{\boldsymbol{\xi} \in \mathcal{D}_\xi} f(\boldsymbol{x}, \boldsymbol{\xi}) \tag{7.2}$$

Preuve.
Soit $\boldsymbol{x}^0 \in \mathcal{D}_x$. L'application $(\boldsymbol{x}, \boldsymbol{\xi}) \longmapsto f\left(\boldsymbol{x}, \boldsymbol{\xi}\right)$ est continue sur $\mathcal{D}_x \times \mathcal{D}_\xi$ qui est un compact. Par le théorème de Heine, la fonction f est donc uniformément continue sur cet ensemble. Nous avons donc :

$$\forall \varepsilon > 0,\ \exists \delta > 0, \forall \boldsymbol{\xi} \in \mathcal{D}_\xi,\ \forall \boldsymbol{x} \text{ tel que } \left\| \boldsymbol{x} - \boldsymbol{x}^0 \right\| \leqslant \delta \Rightarrow \left| f(\boldsymbol{x}, \boldsymbol{\xi}) - f(\boldsymbol{x}^0, \boldsymbol{\xi}) \right| \leqslant \varepsilon$$

Donc $\forall \boldsymbol{\xi} \in \mathcal{D}_\xi$, $\forall \boldsymbol{x}$ tel que $\left\|\boldsymbol{x} - \boldsymbol{x}^0\right\| \leqslant \delta$, on a :

$$\begin{aligned} f(\boldsymbol{x}, \boldsymbol{\xi}) &= f(\boldsymbol{x}, \boldsymbol{\xi}) - f(\boldsymbol{x}^0, \boldsymbol{\xi}) + f(\boldsymbol{x}^0, \boldsymbol{\xi}) \\ &\leqslant \varepsilon + f(\boldsymbol{x}^0, \boldsymbol{\xi}) \\ &\leqslant \varepsilon + g(\boldsymbol{x}^0) \end{aligned}$$

par définition de g pour la dernière ligne. Cette inégalité est vraie $\forall \boldsymbol{\xi} \in \mathcal{D}_\xi$, elle est donc en particulier vraie au point où le maximum est atteint, ce qui nous donne $\forall \boldsymbol{x}$ tel que $\left\|\boldsymbol{x} - \boldsymbol{x}^0\right\| \leqslant \delta$:

$$g(\boldsymbol{x}) \leqslant g(\boldsymbol{x}^0) + \varepsilon \Leftrightarrow g(\boldsymbol{x}) - g(\boldsymbol{x}^0) \leqslant \varepsilon$$

En répétant ce raisonnement en partant de $f(\boldsymbol{x}^0, \boldsymbol{\xi})$ à la place de $f(\boldsymbol{x}, \boldsymbol{\xi})$:

$$\begin{aligned} f(\boldsymbol{x}^0, \boldsymbol{\xi}) &= f(\boldsymbol{x}^0, \boldsymbol{\xi}) - f(\boldsymbol{x}, \boldsymbol{\xi}) + f(\boldsymbol{x}, \boldsymbol{\xi}) \\ &\leqslant \varepsilon + f(\boldsymbol{x}, \boldsymbol{\xi}) \\ &\leqslant \varepsilon + g(\boldsymbol{x}) \end{aligned}$$

par définition de g pour la dernière ligne. Cette inégalité est vraie $\forall \boldsymbol{\xi} \in \mathcal{D}_\xi$, elle est donc en particulier vraie au point où le maximum est atteint, ce qui nous donne $\forall \boldsymbol{x}$ tel que $\left\|\boldsymbol{x} - \boldsymbol{x}^0\right\| \leqslant \delta$:

$$g(\boldsymbol{x}^0) \leqslant g(\boldsymbol{x}) + \varepsilon \Leftrightarrow g(\boldsymbol{x}^0) - g(\boldsymbol{x}) \leqslant \varepsilon$$

Ce qui signifie que pour tout $\boldsymbol{x}^0 \in \mathcal{D}_x$:

$$\forall \varepsilon > 0,\ \exists \delta > 0,\ \forall \boldsymbol{x} \in \mathcal{D}_x \left[\left\|\boldsymbol{x} - \boldsymbol{x}^0\right\| \leqslant \delta \Rightarrow \left|g(\boldsymbol{x}^0) - g(\boldsymbol{x})\right| \leqslant \varepsilon \right]$$

Ce qui prouve la continuité $\forall \boldsymbol{x}^0 \in \mathcal{D}_x$.

□

Remarque sur l'hypothèse de compacité de $\mathcal{D}_x$: Le théorème reste valable même si $\mathcal{D}_x$ n'est pas compact. En effet, dans le cas où $\mathcal{D}_x$ est l'espace $\mathbb{R}^{N_x}$ entier, il suffit de choisir $\boldsymbol{x}^0 \in \mathbb{R}^{N_x}$ et d'appliquer le théorème précédent dans la boule fermée unité centrée en $\boldsymbol{x}^0$ $\mathcal{B}(\boldsymbol{x}^0, 1)$. En appliquant le théorème précédent sur la fonction définie sur $\mathcal{B}(\boldsymbol{x}^0, 1) \times \mathcal{D}_\xi$, on se retrouve de nouveau sur un compact et le théorème précédent est encore valable. Sachant que l'on peut faire cela $\forall \boldsymbol{x}^0 \in \mathbb{R}^{N_x}$, on a bien la continuité (non uniforme) sur tout l'espace $\mathbb{R}^{N_x}$.

Remarque sur le cas où $\mathcal{D}_\xi$ dépend de $\mathcal{D}_x$: il est possible que le domaine de définition de $\boldsymbol{\xi}$ puisse dépendre de $\boldsymbol{x}$. En particulier, les incertitudes portant sur les paramètres d'optimisation $\boldsymbol{\xi}_x$ ont un domaine de variation modifié lorsque le point $\boldsymbol{x}$ se trouve à proximité du bord de son domaine de définition. Ces incertitudes peuvent par exemple être appliquées de la manière suivante : soit le paramètre d'optimisation incertain est de la forme $\boldsymbol{X} = \boldsymbol{x} + \boldsymbol{\xi}$, soit il est de la forme $\boldsymbol{X} = \boldsymbol{x}(1 + \boldsymbol{\xi})$. Dès lors, il est nécessaire de redéfinir le domaine de variation des paramètres incertains $\boldsymbol{\xi}$ pour que $\boldsymbol{X}$ soit défini dans $\mathcal{D}_x$. Afin de prouver que le théorème précédent est encore vrai dans le cas où le domaine de définition de $\boldsymbol{\xi}$ dépend de $\boldsymbol{x}$, il est primordial de ne pas avoir à redéfinir le domaine de variation de $\boldsymbol{\xi}$. À cette fin, une manière équivalente de traduire cette dépendance est d'introduire une fonction continue $\boldsymbol{\zeta}$ définie à l'équation (7.3).

Cette fonction $\boldsymbol{\zeta}$ permet de vérifier que toutes les valeurs prises par $\boldsymbol{\xi}_x$ dans le domaine $\mathcal{D}_{\xi_x}$ correspondent à un $\boldsymbol{X}$ qui se trouve dans $\mathcal{D}_x$. En supposant que la formule permettant de calculer $\boldsymbol{X}$ à partir de $\boldsymbol{x}$ et $\boldsymbol{\xi}$ est continue, la fonction $\boldsymbol{\zeta}$ ainsi définie est également continue. On peut donc se ramener à un problème de la forme :

$$g(x) = \max_{\boldsymbol{\xi} \in \mathcal{D}_\xi} f\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)$$

f et $\boldsymbol{\zeta}$ sont deux fonctions continues. Or, par composition de deux fonctions continues, la fonction $\bar{f}(\boldsymbol{x}, \boldsymbol{\xi}) = f\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)$ est également continue et l'on peut appliquer le théorème précédent.

7.1.2 Choix de l'algorithme d'optimisation mono-objectif pour l'estimation de pire cas

La fonction objectif f_1 avec des entrées incertaines est potentiellement coûteuse à évaluer. De ce fait, l'algorithme d'optimisation qui permet de calculer le pire cas en chaque point $\boldsymbol{x}$ doit converger en peu d'appels à cette fonction. Toutefois, cette caractéristique n'est pas suffisante et un couplage entre l'optimiseur multi-objectif et l'estimation du pire cas doit être mis en place. Pour cela, la section 7.1.2.1 introduit une nouvelle notation permettant de différencier les incertitudes portant sur les paramètres de contrôle $\boldsymbol{x}$ et celles portant sur les paramètres environnementaux. Puis la section 7.1.2.2 détaille ce que cela peut apporter pour diminuer le nombre d'appels à la fonction f_1 lors de la résolution de l'équation (7.1).

7.1.2.1 Réécriture de l'optimisation multi-objectif pire cas

Dans le problème de l'équation (7.1), une partie des incertitudes $\boldsymbol{\xi}_x$ porte sur les paramètres de design $\boldsymbol{x}$. De ce fait, deux valeurs de $\boldsymbol{\xi}_x$ et de $\boldsymbol{x}$ distinctes peuvent correspondre à une même estimation de f_1. Ceci peut notamment permettre de réutiliser des appels à la fonction f_1 utilisés pour estimer le pire cas autour d'un point $\boldsymbol{x}^1$ afin d'estimer le pire cas en un $\boldsymbol{x}^2$ à proximité de $\boldsymbol{x}^1$. Donnons quelques explications, en premier lieu, nous modifions les notations du problème d'optimisation. Soit $\boldsymbol{\zeta}$ la fonction définie de la manière suivante :

$$\begin{array}{rccc} \boldsymbol{\zeta} & : & \mathcal{D}_x \times \mathcal{D}_{\xi_x} & \longrightarrow & \mathcal{D}_x \\ & & (\boldsymbol{x}, \boldsymbol{\xi}_x) & \longmapsto & \boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right) \end{array} \tag{7.3}$$

L'idée est donc de faire apparaître, dans la formulation de la fonction objectif, l'influence d'une partie des incertitudes sur les entrées déterministes $\boldsymbol{x}$. Dans le cas d'incertitudes additives ou relatives, les composantes ζ_i, où $1 \leqslant i \leqslant N_\xi$, sont définies à partir d'une fonction L_i qui est soit de la forme $L_i\left(x_i, \xi_{x_i}\right) = x_i + \xi_i$ ou $L_i\left(x_i, \xi_{x_i}\right) = x_i(1+\xi_i)$. Puisque la sortie de $\boldsymbol{\zeta}$ doit appartenir à $\mathcal{D}_x$, ses composantes sont définies de la manière suivante :

$$\forall i \in \{1, \cdots, N_x\} \quad \zeta_i(x_i, \boldsymbol{\xi}_{x_i}) = \begin{cases} x_i^{min} & \text{si } L_i\left(x_i, \xi_{x_i}\right) \leqslant x_i^{min} \\ x_i^{max} & \text{si } L_i\left(x_i, \xi_{x_i}\right) \geqslant x_i^{max} \\ L_i\left(x_i, \xi_{x_i}\right) & \text{sinon} \end{cases} \tag{7.4}$$

Les notations x_i^{min} et x_i^{max} proviennent du fait que $\mathcal{D}_x$ est un compact de R^{N_x}. Chacune de ses composantes est donc bornée et appartient donc à l'intervalle $\left[x_i^{min}, x_i^{max}\right]$. $\boldsymbol{\zeta}$ correspond donc au maximum (ou au minimum) de deux fonctions régulières donc continues

dans chacune de ses dimensions de sortie. Elle est donc continue et nous pouvons donc appliquer le théorème 7.1 qui prouve la continuité de l'objectif pire cas. Avec cette nouvelle notation, nous pouvons réécrire le problème incertain de la manière suivante :

$$\begin{cases} \min\limits_{x\in\mathcal{D}_x} \max\limits_{\xi\in\mathcal{D}_\xi} \left[f_1(\boldsymbol{\zeta}\left(\boldsymbol{x},\boldsymbol{\xi}_x\right),\boldsymbol{\xi}_e)\right] \\ \min\limits_{x} f_2(\boldsymbol{x}) \end{cases} \tag{7.5}$$

7.1.2.2 Introduction des notations pour la réutilisation des appels à la fonction f_1 entre deux calculs de maximum

Sachant que le pire cas doit être estimé en chaque point de contrôle $\boldsymbol{x}$ proposé par l'algorithme multi-objectif, il est indispensable d'utiliser toutes les techniques possibles afin de diminuer le nombre d'appels nécessaires à la fonction objectif dans la recherche de ces maxima. En plus d'utiliser un algorithme parcimonieux, il peut être intéressant de stocker, pour les réutiliser, les appels réalisés à f_1. En effet, ce pire cas doit être estimé en chaque point de contrôle du problème multi-objectif. Il est donc intéressant d'être capable de réutiliser les appels à la fonction f_1 réalisés pour calculer le maximum autour du point $\boldsymbol{x}^1$ lors du calcul du maximum en un point $\boldsymbol{x}^2$. Plus précisément, pour réaliser cela au cours de l'optimisation multi-objectif, il est nécessaire de stocker tous les appels à la fonction f_1. Soit $\boldsymbol{z}^k = \left(\boldsymbol{\zeta}\left(\boldsymbol{x}^k,\boldsymbol{\xi}_x^k\right),\boldsymbol{\xi}_e^k\right)$ l'entrée du modèle numérique utilisé pour calculer f_1. Introduisons Ξ l'ensemble des appels déjà réalisés à la fonction f_1 :

$$\Xi = \{\boldsymbol{z}\in\mathcal{D}_x\times\mathcal{D}_{\xi_e} \ : \ f_1(\boldsymbol{z}) \text{ a été estimé}\} \tag{7.6}$$

En d'autres termes, Ξ représente la base de données d'appels déjà effectués à f_1 pour calculer le maximum en différents points de contrôle. Cette base de données permet de vérifier s'il y a des simulations déjà exécutées qui peuvent être utilisées afin de déterminer le maximum autour d'un nouveau point $\boldsymbol{x}$. Pour que des points puissent servir dans l'estimation du maximum de f_1 autour de $\boldsymbol{x}$, il faut et il suffit qu'ils appartiennent au sous-ensemble de $\mathcal{D}_x\times\mathcal{D}_{\xi_e}$ noté $\mathcal{D}_{max}\left(\boldsymbol{x}\right)$ défini de la manière suivante :

$$\mathcal{D}_{max}\left(\boldsymbol{x}\right) = \{\boldsymbol{z}'=(\boldsymbol{x}',\boldsymbol{\xi}_e')\in\mathcal{D}_x\times\mathcal{D}_{\xi_e} \ : \ \exists\boldsymbol{\xi}_x\in\mathcal{D}_{\xi_x} \text{ tq } \boldsymbol{\zeta}(\boldsymbol{x},\boldsymbol{\xi})=\boldsymbol{x}'\} \tag{7.7}$$

On remarque que les incertitudes environnementales $\boldsymbol{\xi}_e$ n'influent pas sur la détermination de l'ensemble des points réutilisables.

Ainsi, avant de calculer le maximum en un $\boldsymbol{x}$ quelconque, il suffit de chercher l'ensemble d'appels déjà réalisés et qui se trouvent dans cet ensemble $\mathcal{D}_{max}\left(\boldsymbol{x}\right)$:

$$\Xi_x = \Xi\cap\mathcal{D}_{max}\left(\boldsymbol{x}\right) \tag{7.8}$$

Dans le cas où la fonction $\boldsymbol{\zeta}\left(\boldsymbol{x},\bullet\right)$ est monotone, la recherche des individus de Ξ_x revient donc à garder les points $\boldsymbol{z}'$ de Ξ dont les composantes $\boldsymbol{\zeta}\left(\boldsymbol{x}',\boldsymbol{\xi}_x'\right)$ vérifient N_x inégalités à N_x inconnues du type :

$$\forall i\in\{1,\cdots,N_x\}, \ \min\left(\zeta_i\left(x_i,\xi_{x_i}^{min}\right),\zeta_i\left(x_i,\xi_{x_i}^{max}\right)\right)\leqslant\zeta_i\left(x_i',\xi_{x_i}'\right)\leqslant\max\left(\zeta_i\left(x_i,\xi_{x_i}^{min}\right),\zeta_i\left(x_i,\xi_{x_i}^{max}\right)\right)$$

Ces N_x conditions sont ici faciles à vérifier puisque $\boldsymbol{\zeta}$ se construit à partir de formules simples L_i pour chaque composante, comme défini à l'équation (7.4).

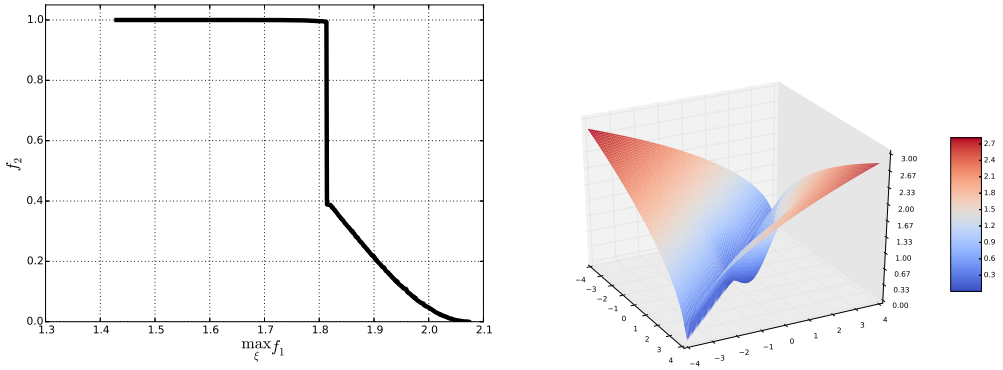


FIGURE 7.1 – Illustration du $PF^\star$ dans le cas où $N_x = 4$ et $N_\xi = 8$ et de l'objectif incertain dans le cas où $N_x = 2$ et $N_{\xi_e} = 0$

Remarque : même si les incertitudes portant sur des variables environnementales $\boldsymbol{\xi}_e$ n'influent pas sur la possibilité de réutiliser un point, elles ont leur importance dans le cas où le nombre d'appels réutilisables est grand et qu'il faut choisir les plus adaptés parmi eux. Ceci est détaillé dans la section 7.1.3.

7.1.2.3 Cadre de l'optimisation mono-objectif

Pour résumer, l'algorithme d'optimisation mono-objectif utilisé pour le calcul du maximum en chaque point de contrôle doit :

1. converger en peu d'appels à la fonction objectif vers le maximum,
2. permettre la réutilisation des appels à la fonction objectif entre deux calculs de maximum.

Ces deux points sont indispensables dans le but de minimiser le nombre d'appels nécessaires à la fonction f_1 afin de converger vers $PF^\star$.

Remarque sur la réutilisation des appels : cette réutilisation repose sur le fait que certaines incertitudes portent sur les paramètres d'entrée. Dans le cas où les incertitudes sont entièrement indépendantes du point $\boldsymbol{x}$, cette contrainte pour le choix de l'algorithme n'a pas de sens. Ce n'est toutefois pas le cas ici et ce critère a son importance dans le choix de la méthode la plus adaptée.

7.1.2.4 Cas test analytique

Afin d'éprouver l'algorithme pire cas sur un cas test d'une complexité suffisante, nous reprenons le cas test MOP2 du chapitre précédent. Toutefois, de sorte à obtenir un front de Pareto non trivial, nous devons modifier l'un des deux objectifs. En effet, sans cette modification, le calcul du maximum sur l'une des fonctions du cas MOP2 sature autour de la valeur 1. La modification introduite ici permet ainsi d'obtenir un front d'une forme non triviale, comme représenté sur la figure 7.1.

$$\begin{cases} f_1(\boldsymbol{x}, \boldsymbol{\xi}_e) = \sqrt{|x_1 + x_2|} \left(1 - \exp\left[-\sum_{i=1}^{N_x} \left(x_i - \frac{1}{\sqrt{N_x + N_{\xi_e}}}\right)^2 - \sum_{i=1}^{N_{\xi_e}} \left(\xi_{e_i} - \frac{1}{\sqrt{N_x + N_{\xi_e}}}\right)^2\right]\right) \\ f_2(\boldsymbol{x}) = 1 - \exp\left[-\sum_{i=1}^{N_x} \left(x_i + \frac{1}{\sqrt{N_x}}\right)^2\right] \end{cases} \tag{7.9}$$

Ce cas test est utilisé avec les dimensions du problème industriel, ce qui signifie $N_x = 4$ et $N_\xi = 8$. Le domaine de variation est le suivant : $\forall i \in \{1, \cdots, N_x\}$, $-4 \leqslant x_i \leqslant 4$. En ce qui concerne les incertitudes $\boldsymbol{\xi}_x$ portant sur les paramètres de contrôle, elles sont de type additive. Ceci revient à définir la fonction $\boldsymbol{\zeta}$ de la manière suivante : $\zeta_i(x_i, \xi_i) = x_i + \xi_i$ pour toutes les composantes i de N_{ξ_x}. Enfin, pour le domaine de variation des incertitudes, nous avons :

$$\text{pour } \boldsymbol{\xi}_x \ : \begin{cases} -2.5 \leqslant \xi_{x_1} \leqslant 2.5 \\ -0.8 \leqslant \xi_{x_2} \leqslant 0.8 \\ -0.9 \leqslant \xi_{x_3} \leqslant 0.9 \\ -0.01 \leqslant \xi_{x_4} \leqslant 0.01 \end{cases} \qquad \text{pour } \boldsymbol{\xi}_e \ : \begin{cases} -0.15 \leqslant \xi_{e_1} \leqslant 0.15 \\ -1.2 \leqslant \xi_{e_2} \leqslant 1.2 \\ -0.125 \leqslant \xi_{e_3} \leqslant 0.125 \\ -3.0 \leqslant \xi_{e_1} \leqslant 3.0 \end{cases}$$

Remarque concernant le domaine de variation de la sortie de $\boldsymbol{\zeta}$: avec les définitions données précédemment pour le domaine de variation de $\boldsymbol{x}$ et de $\boldsymbol{\xi}_x$, il est possible que la somme $x_i + \xi_{x_i}$ sorte du domaine de variation de $\boldsymbol{x}$. Comme expliqué dans la section 7.1.1, la fonction $\boldsymbol{\zeta}$ est définie de manière à ce que sa sortie sature dans le domaine de définition $\mathcal{D}_x$.

Remarque concernant l'estimation du $PF^\star$ de la figure 7.1 : l'introduction des incertitudes dans le problème analytique rend la connaissance de la solution analytique plus ardue. Afin d'avoir une référence à laquelle se comparer, nous avons appliqué sur le cas test MOP2 modifié l'algorithme NSGA-II (détaillé en section 5.2.1) couplé à l'algorithme à évolution différentielle (section 5.1.2.4) afin de calculer le maximum de f_1 en chaque $\boldsymbol{x}$. En répétant 50 fois cette optimisation et en gardant les points Pareto-optimaux de ces 50 répétitions, nous avons une approximation suffisamment précise de $PF^\star$.

7.1.2.5 Comparaison des estimateurs pire cas

Dans cette partie, nous comparons trois algorithmes d'optimisation mono-objectif dans le but de choisir celui remplissant au mieux les conditions énoncées dans la section 7.1.2.3. Les trois algorithmes à comparer que nous avons choisis sont les suivants :

1. L'algorithme à évolution différentielle, expliqué à la section 5.1.2.4. C'est un algorithme évolutionnaire (donc stochastique) qui présente l'avantage d'avoir peu de paramètres à fixer. La librairie Scipy [Jones *et al.*, 01] en fournit une version implémentée. Les paramètres par défaut sont utilisés.
2. L'algorithme DIRECT, présenté dans la section 5.1.2.3. C'est un algorithme échantillonnant l'espace des entrées $\mathcal{D}_x$ de manière déterministe, c'est-à-dire que la recherche du minimum est basée sur des critères déterministes. L'utilisation de cet algorithme se fait par la bibliothèque NLOPT [Johnson, 2011], utilisable sous Python. Ici encore, les paramètres par défaut sont utilisés.

Algorithme d'optimisation	**EGO**	**DIRECT**	**Évolution Différentielle**
Pourcentage d'échec (erreur relative $> 10^{-2}$)	0.0	0.0	0.0
Nombre d'appels moyen à f_1	25	450	500

TABLEAU 7.1 – Résultat de la résolution de $\max_{\xi \in \mathcal{D}_\xi} \left[f_1(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e) \right]$ sur le cas MOP2 modifié de l'équation (7.9) en 100 points $\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{100}$ tirés par LHS. L'optimisation est répétée dix fois en chaque paramètre de contrôle.

3. L'algorithme EGO, présenté dans la section 5.1.3.1. Ce dernier a été implémenté en Python. L'unique bibliothèque utilisée ici est Scikit-learn [Pedregosa *et al.*, 2011], permettant en particulier de construire des modèles de krigeage en Python. Pour la maximisation du critère EI, l'algorithme à évolution différentielle est utilisé.
   Pour minimiser le nombre d'appels nécessaires à f_1, le critère d'arrêt utilisé ne porte pas uniquement sur un nombre d'appels maximal. En plus de ce budget maximal, nous utilisons également des critères d'arrêt portant sur la valeur de l'amélioration espérée au cours des itérations. Plus précisément, et puisque l'amélioration espérée (détaillée à l'équation (5.3)) peut beaucoup varier d'une itération à une autre, le critère porte sur la moyenne glissante des cinq dernières valeurs prises par le maximum de l'EI :
   $$\bar{EI}^k = \frac{1}{5} \sum_{j=0}^{4} EI^{k-j} \tag{7.10}$$
   Ici, k correspond à l'itération à laquelle nous nous trouvons. En d'autres termes, il traduit le nombre de points déjà rajoutés au DOE par maximisation de l'EI. Les deux critères d'arrêt de l'algorithme EGO utilisés sont donc les suivants :
   $$\begin{gathered} \bar{EI}^k < \varepsilon_f \min_{\boldsymbol{x} \in DOE^k} f(\boldsymbol{x}) \\ \text{ou} \\ \frac{\bar{EI}^k - \bar{EI}^{k-1}}{\bar{EI}^{k-1}} < \varepsilon_{EI} \end{gathered} \tag{7.11}$$
   Dans les applications présentées, nous prenons $\varepsilon_f = \varepsilon_{EI} = 10^{-3}$. Nous utilisons un plan d'expériences initial avec un point par dimension, donc composé de 8 points.

Afin de choisir l'algorithme le plus adapté, nous appliquons chacun d'eux au cas test analytique introduit et au cas test industriel. Pour cela, nous réalisons le calcul du maximum autour de 100 points $\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{100}$ tirés dans $\mathcal{D}_x$ aléatoirement par LHS (détaillé à la section 3.1.3). Puisque deux de ces algorithmes sont stochastiques, il est nécessaire de répéter l'optimisation plusieurs fois autour d'un même jeu de paramètre de contrôle afin d'en vérifier la convergence. Ici, l'optimisation est répétée 10 fois pour chacun de ces jeux de paramètres.

Le résultat est donné sur les tableaux 7.1 pour le cas MOP2 modifié et 7.2 pour le cas du problème industriel. L'algorithme EGO est celui qui répond le mieux à notre cadre, c'est-à-dire aux deux propriétés développées à la section 7.1.2.3 :

1. D'abord, comme le soulignent les deux tableaux de résultat, EGO nécessite moins d'appels à f_1 pour atteindre la convergence avec une précision en 10^{-2}.

Algorithme d'optimisation	**EGO**	**DIRECT**	**Évolution Différentielle**
Pourcentage d'échec (erreur relative $> 10^{-2}$)	0.0	0.0	0.0
Nombre d'appels moyen à f_1	38	355	683

TABLEAU 7.2 – Résultat de la résolution de $\max_{\xi \in \mathcal{D}_\xi} \left[f_1(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e) \right]$ sur le cas test industriel en 100 points $\boldsymbol{x}^1, \cdots, \boldsymbol{x}^{100}$ tirés par LHS. L'optimisation est répétée dix fois en chaque paramètre de contrôle.

2. Ensuite, EGO permet très aisément de prendre en compte les appels déjà réalisés à f_1 (enjeu expliqué à la section 7.1.2.2) à la différence des deux autres méthodes testées. DIRECT échantillonne l'espace de manière déterministe, il n'est donc pas possible de facilement initialiser la méthode à l'aide des points de l'ensemble Ξ_x (défini à l'équation (7.8)). En ce qui concerne l'algorithme à évolution différentielle, les simulations de Ξ_x peuvent être intégrées à la population initiale. Toutefois, cette population initiale est réduite et de taille fixée. Elle est de plus la base de l'exploration de l'espace de design que permet cet algorithme. Or, rien ne garantit que les points de Ξ_x soient bien répartis. Leur utilisation peut donc ralentir la convergence, par exemple par rapport à une initialisation aléatoire.

   Enfin, dans le cas de EGO, les points de Ξ_x peuvent simplement être utilisés dans le plan d'expériences initial. De plus, ce plan d'expériences initial n'étant théoriquement pas limité en taille, on peut le compléter de quelques points pour remplir l'espace au mieux et ainsi éviter les désagréments liés à une mauvaise exploration initiale. Ceci est expliqué dans la section suivante. L'idée est de ne pas favoriser à tort une zone de l'espace plutôt qu'une autre dans le plan d'expériences initial.

Remarque : l'algorithme EGO est celui qui répond le mieux aux besoins que nous avons. Toutefois, pour obtenir une précision plus importante sur le minimum, par exemple une erreur relative de l'ordre de 10^{-5}, il peut être préférable d'utiliser l'algorithme DIRECT. En effet, EGO est basé sur l'enrichissement du plan d'expériences d'un modèle de krigeage. Or, pour obtenir des précisions de cet ordre, l'algorithme peut nécessiter de rajouter un grand nombre de points dans une zone très réduite de l'espace, ce qui mène à un mauvais conditionnement de la matrice de covariance. Or cette matrice de covariance doit être inversée pour calculer les paramètres optimaux ainsi que pour utiliser le modèle en prédiction. Ce mauvais conditionnement peut rendre l'inversion impossible. Ceci peut donc mener à l'arrêt prématuré de la méthode dans le cas où plus de précision est souhaité. Pour résoudre cela, l'étude de Mohammadi et ses coauteurs [Mohammadi *et al.*, 2015] propose par exemple d'utiliser le résultat de EGO pour initialiser CMA-ES, qui permet d'intensifier la recherche et donc d'améliorer la précision du résultat retourné.

7.1.3 Réutilisation des appels à la fonction f_1 entre deux calculs de maximum et mise en place de l'algorithme multi-objectif pire cas

En section 6.4, nous avons proposé puis validé un algorithme de résolution de problème multi-objectif nécessitant peu d'appels distribués pour atteindre la convergence. Ensuite

à la section 7.1.1, nous avons prouvé que la fonction objectif $\max_{\xi \in \mathcal{D}_\xi} [f_1(\boldsymbol{\zeta}(\boldsymbol{x}, \boldsymbol{\xi}_x), \boldsymbol{\xi}_e)]$ était continue. Ceci justifie l'utilisation de l'algorithme multi-objectif précédent.

Par ailleurs, nous avons choisi un algorithme d'estimation de pire cas qui répond à notre cadre. Il nécessite peu d'appels à la fonction de référence et il permet de réutiliser des exécutions de f_1 déjà réalisées en les rajoutant au plan d'expériences initial du modèle de substitution.

De ce fait, l'unique étape manquante afin d'avoir un algorithme d'optimisation pire cas *complet* est de détailler la manière de réutiliser les points dans le cadre de l'optimiseur multi-objectif.

Nous expliquons ici le choix des appels à f_1 déjà réalisées de Ξ (défini à l'équation (7.8)) que nous utilisons pour le calcul du maximum de f_1 en un point $\boldsymbol{x} \in \mathcal{D}_x$. Ceci est détaillé à l'algorithme 7.1. L'idée de cet algorithme est la suivante : nous utilisons un plan d'expériences initial de taille minimale $N^{min}_{DOE_0}$, et maximale de taille $N^{max}_{DOE_0}$. Pour générer ce DOE initial, il faut d'abord vérifier si des individus de Ξ se trouvent dans $\mathcal{D}_{max}(\boldsymbol{x})$ (ce qui correspond à l'image de la fonction $\boldsymbol{\zeta}(\boldsymbol{x}, \bullet)$). Avec nos notations, ceci est équivalent à vérifier la condition suivante : $\text{card}(\Xi_x) > 0$. Si elle n'est pas vérifiée, nous tirons $N^{min}_{DOE_0}$ points aléatoirement. Si cette assertion est vérifiée, il est important de garder les points de Ξ_x qui remplissent au mieux l'espace des entrées. Afin de s'assurer que le plan d'expériences initial ne soit pas lacunaire, on se donne la possibilité de simuler de nouveaux points pour construire le DOE initial, en particulier si trop de points de Ξ_x se trouvent dans la même zone de l'espace $\mathcal{D}_{max}(\boldsymbol{x})$. Tout d'abord, si $\text{card}(\Xi_x) < N^{max}_{DOE_0} - N^{min}_{DOE_0}$, nous prenons tous les points de Ξ_x et en simulons $N^{min}_{DOE_0}$ pour avoir un plan d'expériences au moins aussi bien réparti que dans le cas où nous ne réutilisons pas de points. Si $\text{card}(\Xi_x) \geqslant N^{max}_{DOE_0}$, puisque nous recherchons le maximum de f_1, nous rajoutons automatiquement l'individu de Ξ_x qui maximise f_1. Ensuite, pour ne pas orienter à tort la recherche menée par EGO vers des zones non alimentées en points dans le DOE, nous proposons de réutiliser l'idée d'un plan d'expériences LHS (détaillé à la section 3.1.3). C'est-à-dire que nous décomposons chaque dimension de $\mathcal{D}_{max}(\boldsymbol{x})$ en $N^{max}_{DOE_0}$ sous-domaines. Pour continuer à donner de l'importance aux appels déjà réalisés maximisant f_1, les points de Ξ_x maximisant f_1 dans chacune des sous-lignes non vides sont gardés. Enfin, pour obtenir le nombre de point final recherché, des points tirés aléatoirement dans l'espace d'entrée $\mathcal{D}_{max}(\boldsymbol{x})$ sont évalués sur la fonction f_1. À la manière d'un LHS (détaillé à la section 3.1.3), ces derniers sont choisis de manière aléatoire et de sorte à remplir au mieux l'espace par rapport aux points déjà sélectionnés, comme à la figure 3.7. On utilise pour cela la décomposition en $\left(N^{max}_{DOE_0}\right)^{N_x + N_{\xi_e}}$ *cubes* et on fait en sorte de rajouter des points se trouvant dans des cubes où les colonnes selon chaque dimension sont vides de points.

7.1.4 Application sur le cas analytique

L'algorithme MOEGO NSGA-II (détaillé en section 6.4) avec la réutilisation détaillée à l'algorithme 7.1 est mis en place sur le cas test MOP2 modifié proposé dans la partie 7.1.2.4. Pour le calcul du pire cas sur l'objectif f_1, nous utilisons un algorithme EGO (détaillé à l'algorithme 5.2). Les paramètres de l'algorithme multi-objectif sont les mêmes qu'au chapitre précédent et sont résumés à la figure 6.37. Pour l'EGO mono-objectif, on utilise les critères d'arrêt de l'équation (7.11) et détaillés à la section 7.1.2.5. Les paramètres de l'algorithme EGO sont les mêmes que dans cette dernière section avec $\varepsilon_f = \varepsilon_{EI} = 10^{-3}$. Pour l'algorithme 7.1 de réutilisation le plan d'expériences initial de

Algorithme 7.1 Algorithme de réutilisation des appels

Input : Ξ la base de données d'appels déjà réalisés et appartenant à $\mathcal{D}_{max}(\boldsymbol{x})$
$\boldsymbol{x} \in \mathcal{D}_x$ le point courant où l'on souhaite évaluer le maximum
La taille maximale souhaitée pour le DOE initial $N_{DOE_0}^{max}$
La taille minimale souhaitée pour le DOE initial $N_{DOE_0}^{min}$

Output : $\boldsymbol{Z}^0$, $\boldsymbol{Y}_z^0$ le plan d'expériences initial pour le calcul du maximum en $\boldsymbol{x}$ par EGO

· Rechercher les $\boldsymbol{z} \in \Xi_x$ (défini à l'équation (7.8));
if $\text{card}(\Xi_x) == 0$ **then**
· Générer les $N_{DOE_0}^{min}$ jeux d'entrées $\boldsymbol{Z}^0$ par LHS;
· Évaluer $\boldsymbol{Y}_z^0 = f_1(\boldsymbol{Z}^0)$;
if $1 \leqslant \text{card}(\Xi_x) \leqslant N_{DOE_0}^{max} - N_{DOE_0}^{min}$ **then**
· Générer les $N_{DOE_0}^{min}$ jeux d'entrées $\boldsymbol{Z}^0$ par LHS;
· Évaluer $\boldsymbol{Y}_z^0 = f_1(\boldsymbol{Z}^0)$;
· $\boldsymbol{Z}^0 = \left\{\boldsymbol{Z}^0, \Xi_x\right\}$;
· $\boldsymbol{Y}_z^0 = \left\{\boldsymbol{Y}_z^0, f_1(\Xi_x)\right\}$;
else
· Décomposer $\mathcal{D}_{max}(\boldsymbol{x})$ par une grille de taille $\left(N_{DOE_0}^{max}\right)^{N_x+N_{\xi_e}}$;
for $1 \leqslant i \leqslant N_x + N_{\xi_e}$ **do**
· Calculer le nombre de colonnes avec des points le long de la dimension i $\rightarrow N_c(i)$;
end
· Garder la dimension $d = \text{argmax}_{1 \leqslant i \leqslant N_x+N_{\xi_e}} N_c(i)$;
· Construire $(S_k)_{1 \leqslant k \leqslant N_{DOE_0}^{max}}$ défini telle que :

$$S_k = \left\{ \boldsymbol{z} \in \Xi_x \ : \ z_d^{min} + k\frac{z_d^{max} - z_d^{min}}{N_{DOE_0}^{max}} \leqslant z_d \leqslant z_d^{min} + (k+1)\frac{z_d^{max} - z_d^{min}}{N_{DOE_0}^{max}} \right\}$$

· $\boldsymbol{Z}^0 = \emptyset$;
for $1 \leqslant k \leqslant N_{DOE_0}^{max}$ **do**
· Si $S_k \neq \emptyset$, $\boldsymbol{Z}^0 = \left\{\boldsymbol{Z}^0, \text{argmax}_{z \in S_k} f_1(\boldsymbol{z})\right\}$;
end
if $N_c(d) < N_{DOE_0}^{max}$ **then**
· Générer les $N_{DOE_0}^{max} - N_c(d)$ jeux d'entrées $\boldsymbol{Z}^{temp}$ aléatoirement dans les colonnes vides S_k;
· Vérifier que les colonnes selon les autres dimensions en ces points sont vides;
· Évaluer ces points sur f_1 : $\boldsymbol{Y}_z^{temp} = f_1\left(\boldsymbol{Z}^{temp}\right)$;
· $\boldsymbol{Z}^0 = \left\{\boldsymbol{Z}^0, \boldsymbol{Z}^{temp}\right\}$;
end
· Retourner $\boldsymbol{Z}^0$, $\boldsymbol{Y}_z^0$;

ces EGO mono-objectifs est pris de taille minimum $N_{DOE_0}^{min} = 8$ si Ξ_x est vide et de taille maximum $N_{DOE_0}^{max} = 40$ si Ξ_x n'est pas vide. Par exemple en dimension 8, ceci équivaut à 5

points par dimension. De plus, les points rajoutés, dans le cas où $\text{card}\,(\Xi_x) > 8$ sont pour la plupart des points déjà estimés. Leur ajout peut donc être considéré comme *gratuit*.

7.1.4.1 Efficacité de MOEGO NSGA-II couplé à EGO

Afin de valider notre approche, nous comparons ici le résultat de MOEGO NSGA-II couplé à EGO au résultat de l'algorithme NSGA-II classique pour la partie multi-objectif de l'équation (7.1) et l'algorithme à évolution différentielle pour estimer le pire cas en chaque point de contrôle multi-objectif $\boldsymbol{x}$. Cette résolution est volontairement lourde et coûteuse en nombre d'appels aux deux fonctions objectif. Le résultat de cette optimisation constitue la référence à laquelle se comparer.

Comme précédemment, nous répétons l'optimisation cinquante fois et nous comparons la décroissance de la distance entre $PF^\star$ discrétisé et le front retourné par chacune des ces deux méthodes. La figure 7.2 résume ce résultat. La distance cible choisie ici correspond à $\mathcal{M}^{target} = 0.005$, où la distance est celle détaillée à l'équation (6.6). La figure 7.3 illustre un front obtenu lorsque cette distance cible est atteinte. On remarque que notre méthodologie nécessite un nombre restreint d'appels aux fonctions objectif afin d'atteindre une précision suffisante sur le front retourné. En particulier, la figure 7.4 met en évidence l'itération à partir de laquelle toutes les répétitions de l'algorithme ont atteint la distance cible. On remarque que MOEGO NSGA-II permet de converger en environ 39 appels distribués, alors que NSGA-II couplé à l'évolution différentielle (pour le pire cas) en nécessite 105.

Nous souhaitons ensuite mettre en évidence l'apport de la réutilisation des appels à f_1 détaillée à l'algorithme 7.1. Pour cela, la figure 7.5 montre le nombre moyen d'appels à la fonction f_1 à chaque calcul de pire cas et pour chaque itération de l'algorithme MOEGO NSGA-II. Cette figure montre que la réutilisation a un effet positif sur le nombre d'appels nécessaires à la fonction f_1, puisque plus le nombre d'itérations est important, donc plus la base de données Ξ est remplie, moins il est nécessaire d'exécuter de simulations de f_1. Par ailleurs, ce graphique illustre, comme la figure 6.47 au chapitre précédent, que MOEGO NSGA-II a une première phase à nombre d'appels réduit, une seconde phase où le nombre d'appels augmente beaucoup puis une phase finale qui nécessite moins d'appels. La zone où le nombre d'appels augmente traduit le fait que MOEGO NSGA-II explore de manière importante l'espace $\mathcal{D}_x$. Ceci implique que les points à évaluer sont distants et donc que le cardinal de Ξ_x est nul. En d'autres termes, qu'il n'y aucun point de la base de données Ξ qui se trouve à proximité des points de contrôle où l'on évalue le pire cas à ces itérations. Ceci met également en évidence que lors des premières itérations de l'algorithme MOEGO NSGA-II, l'exploration ne se met pas tout de suite en place et la méthode stagne dans une zone, ce qui justifie l'efficacité de la réutilisation.

Enfin, la fonction f_1 peut être coûteuse. Or, nous lui appliquons le calcul d'un pire cas à chaque appel distribué. Pour quantifier le coût de résolution de l'équation (7.1), il faut regarder le nombre moyen d'appels nécessaires à cette fonction à chaque itération de l'algorithme MOEGO NSGA-II. La figure 7.6 donne le nombre maximum d'appels à chaque itération de MOEGO NSGA-II. De cette dernière, on peut en déduire le temps d'exécution d'un tel algorithme. En effet, on remarque sur ce graphique qu'il faut une moyenne de 36 exécutions de f_1 au maximum afin de calculer le pire cas, en particulier grâce à l'algorithme 7.1 de réutilisation. De ce fait, en conjuguant les 39 itérations de MOEGO NSGA-II (donnée extraite de la figure 7.4) et les 36 exécutions de f_1 à chaque itération, le temps d'exécution de l'algorithme est de l'ordre de 1400 exécutions de la fonction f_1 pour

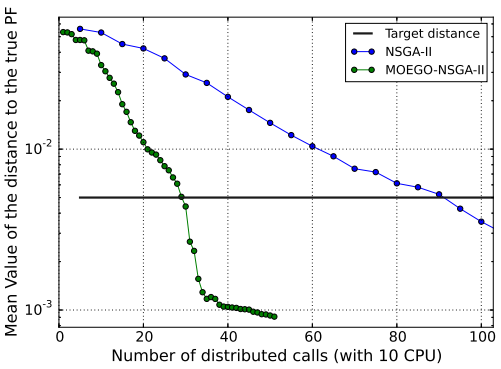


FIGURE 7.2 – Graphique représentant la moyenne sur cinquante optimisations de la distance entre $PF^\star$ et PF en fonction du nombre d'appels distribués aux fonctions objectif pour le cas test MOP2 modifié

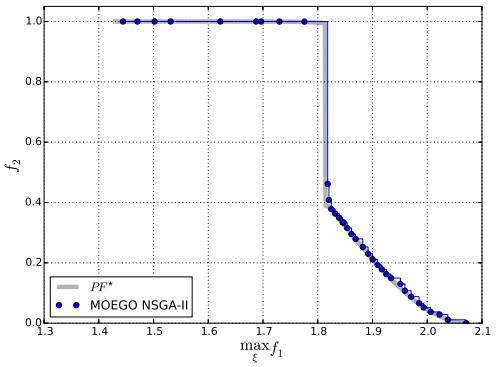


FIGURE 7.3 – Graphique représentant le front obtenu par MOEGO NSGA-II après 30 appels distribués et vérifiant le critère de l'équation (6.8) avec $\mathcal{M}^{target} = 0.005$.

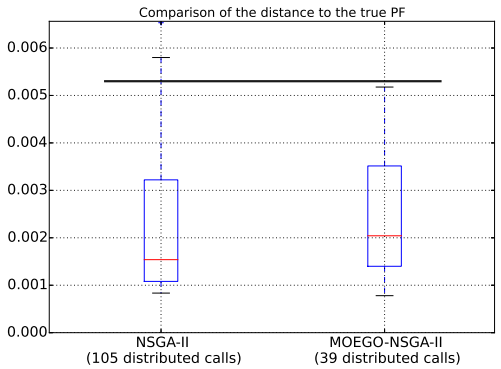


FIGURE 7.4 – Boîtes à moustaches du nombre d'appels aux fonctions objectif une fois le critère d'arrêt atteint pour le cas test MOP2 modifié

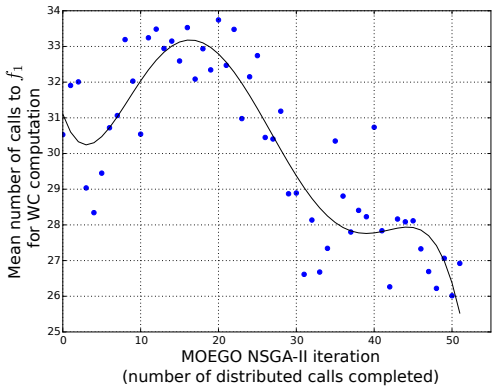


FIGURE 7.5 – Nombre moyen d'appels à f_1 à chaque calcul de pire cas pour chaque itération de l'algorithme MOEGO NSGA-II

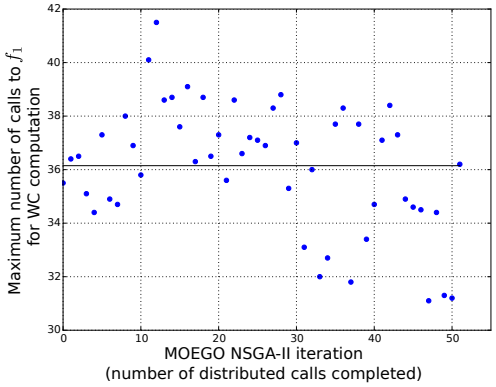


FIGURE 7.6 – Nombre maximum d'appels à f_1 à chaque calcul de pire cas pour chaque itération de l'algorithme MOEGO NSGA-II

atteindre la convergence. Cette estimation se base sur la pire des cinquante répétitions réalisées. Ce qui nous donne donc une borne supérieure du nombre d'appels nécessaires.

7.1.4.2 Preuve de l'intérêt de l'optimisation pire cas

Un autre aspect qu'il est intéressant de mettre en évidence ici est l'apport de la prise en compte des incertitudes sur les entrées d'un problème d'optimisation.

Pour cela, nous comparons les solutions des deux formulations introduites jusqu'à maintenant. La première est celle ne prenant pas en compte les incertitudes sur les différents paramètres du problème d'optimisation, c'est-à-dire celle donnée à l'équation (6.1). La seconde est celle de l'équation (7.1) où les incertitudes sont prises en compte par un pire cas en chaque point de contrôle.

Nous effectuons cette comparaison sur le cas test introduit à l'équation (7.9). La figure 7.8 met sur un même graphique le front solution de l'équation (6.1), ici appelé déterministe, et le front solution de l'équation (7.1), ici appelé pire cas. On remarque que la prise en compte des incertitudes par le pire cas modifie la forme du front optimal. Plus précisément, si l'on calcule à présent le pire cas sur l'objectif f_1 des solutions Pareto-optimales déterministes (ce que représente la figure 7.8), on peut mettre en évidence deux points importants :

1. D'abord, les solutions du problème déterministe restent Pareto-optimales pour la formulation pire cas de l'équation (7.1).
2. Ensuite, un grand nombre de solutions qui n'étaient pas Pareto-optimales pour la formulation déterministe le deviennent dans la formulation prenant en compte les incertitudes par un pire cas.

Le deuxième point est important puisque les incertitudes peuvent traduire des effets non maîtrisables par l'utilisateur, telles que des contraintes d'usinage. De ce fait, ces solutions introduites par la résolution de la formulation pire cas présentent l'avantage d'être Pareto-optimales et robustes. La non-prise en compte des incertitudes présente le désavantage dans notre cas de ne pas saisir certaines solutions qui peuvent être d'intérêt pour l'utilisateur.

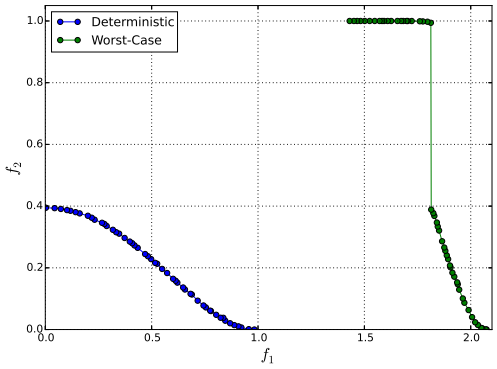


FIGURE 7.7 – Représentation dans l'espace des objectifs des points Pareto-optimaux pour la formulation déterministe et pour la formulation avec un pire cas pour quantifier les effets des incertitudes en entrée.

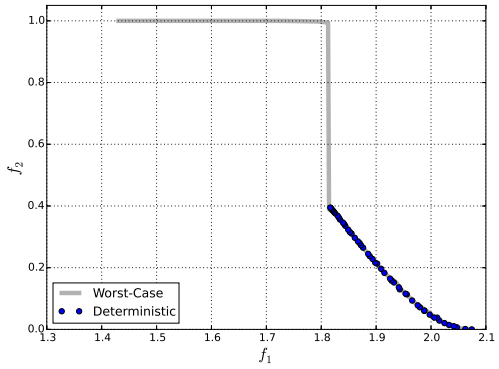


FIGURE 7.8 – Représentation du $\max_{\boldsymbol{\xi}} f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)$ en fonction de f_2 des points Pareto-optimaux pour la formulation déterministe et pour la formulation avec un pire cas pour quantifier les effets des incertitudes en entrée.

7.2 Application à l'optimisation pire cas sur le problème industriel

Le problème industriel est présenté au chapitre 2. En particulier, le problème d'optimisation à résoudre est donné à l'équation (2.12). Avec les notations de cette partie, notamment l'introduction de la fonction $\boldsymbol{\zeta}$ (à l'équation (7.3)), nous pouvons le reformuler de la manière suivante :

$$\begin{cases} \min\limits_{x} \max\limits_{\xi} f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right) \\ \min\limits_{x} f_2(\boldsymbol{x}) \end{cases} \tag{7.12}$$

Le problème industriel est décrit en détail au chapitre 2. Pour rappel, la fonction f_2 est analytique et peu coûteuse puisqu'elle correspond à la masse et au coût liés au choix d'une solution de refroidissement donnée. La fonction f_1 est quant à elle potentiellement coûteuse. Elle correspond ici au calcul de perte de durée de vie par rapport à l'équipement de référence. Pour rappel, il s'agit d'un équipement déjà existant dans un avion de référence donné. Ce calcul nécessite l'exécution de simulations thermiques transitoires. Ici,

ces simulations proviennent d'un modèle nodal fourni par Epsilon. Comme cela est précisé à la section 2.2.1.1 présentant ce type de modèle, cette modélisation est précise dans le cas où il n'y a pas de changement de type de convection. En particulier le passage de la convection naturelle à la convection forcée est une source d'erreur importante. Or, nous ne devons calculer ici la durée de vie que dans un cadre nominal d'utilisation. En effet, nous souhaitons estimer le nombre de vols en fonctionnement **nominal** supportable par l'équipement électronique (ceci est expliqué dans la section 2.2.2). Le modèle thermique instationnaire n'a donc pas vocation à être utilisé pour prédire le passage d'un régime de convection à un autre. De ce fait, puisque nous avons besoin d'exécuter des simulations thermiques transitoires qui se situent dans le domaine d'utilisation dans lequel a été validé un modèle nodal, ce dernier est adapté à la résolution de l'équation (7.12). Ce type de modèle a un temps d'exécution de l'ordre de la minute. MOEGO NSGA-II couplé à EGO pour l'estimation de pire cas est donc applicable au cas industriel.

7.2.1 Résultat de l'optimisation multi-objectif pire cas

Comme pour le cas analytique, nous exécutons l'algorithme MOEGO NSGA-II (détaillé en section 6.4) avec la réutilisation détaillée à l'algorithme 7.1. Les paramètres pour l'algorithme MOEGO NSGA-II sont les mêmes qu'à la figure 6.37. Pour le calcul du pire cas sur l'objectif f_1, nous utilisons un algorithme EGO (détaillé à l'algorithme 5.2) avec les critères d'arrêt de l'équation (7.11) et détaillés à la section 7.1.2.5. Les paramètres de l'algorithme EGO sont les mêmes que dans cette dernière section avec $\varepsilon_f = \varepsilon_{EI} = 10^{-3}$. Le plan d'expériences initial de ces EGO mono-objectifs est au minimum de taille $N_{DOE_0}^{min} = 8$ et au maximum de taille $N_{DOE_0}^{max} = 40$. Pour rappel, bien que soit de dimension quatre, c'est-à-dire $\boldsymbol{x} \in \mathcal{D}_x \subset \mathbb{R}^4$, les incertitudes sont de dimension huit, donc $\boldsymbol{\xi} \in \mathcal{D}_\xi \subset \mathbb{R}^8$.

La figure 7.9 compare le front obtenu après 30 itérations et 50 itérations. On remarque que la convergence est atteinte entre ces deux itérations puisqu'il y a très peu de variabilité d'une itération à une autre. Si l'on s'intéresse au nombre d'appels nécessaires à f_1 afin de converger, la figure 7.10 montre que chaque session d'appels distribués nécessite en moyenne un maximum de 36 appels à la fonction f_1. Le calcul est donc le même que pour le cas test analytique et le temps d'exécution total de l'algorithme est de l'ordre de $30 \times 36 \simeq 1000$ appels à la fonction f_1.

7.2.2 Comparaison entre le front pire cas et le front déterministe

Comme pour l'application de la section 7.1.4, nous quantifions dans cette partie l'intérêt d'avoir pris en compte les incertitudes au niveau des paramètres de contrôle $\boldsymbol{x}$ des solutions obtenues. Pour cela, nous comparons les solutions des deux formulations introduites jusqu'à maintenant. La première est celle ne prenant pas en compte les incertitudes sur les différents paramètres du problème d'optimisation, c'est-à-dire celle donnée à l'équation (6.1). La seconde est celle de l'équation (7.1) où les incertitudes sont prises en compte par un pire cas en chaque point de contrôle.

La figure 7.11 illustre d'abord l'écart qu'il y a entre la solution ne prenant pas en compte les incertitudes sur les entrées et celle les quantifiant par un pire cas. Cela permet de mettre en évidence le fait que la distance entre les solutions Pareto-optimales pour le pire cas et les solutions Pareto-optimales de la formulation déterministe n'est pas constante quelle que soit la valeur de f_2. En effet, sur la partie gauche du front où $f_2 < 0.3$, on remarque que l'écart entre ces deux ensembles de point est pratiquement constant. Toutefois, si l'on

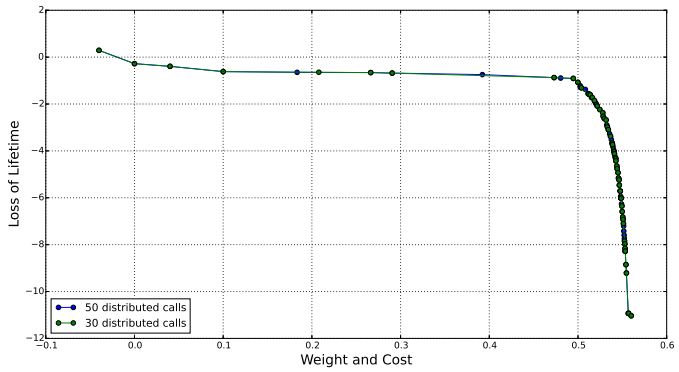


FIGURE 7.9 – Comparaison du front de Pareto pire cas obtenu par MOEGO NSGA-II couplé avec EGO après 30 itérations de MOEGO NSGA-II et après 50 itérations de MOEGO NSGA-II.

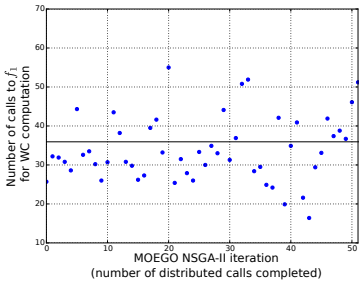


FIGURE 7.10 – Nombre maximal d'appels à f_1 réalisés par l'algorithme EGO à chaque itération de l'algorithme MOEGO NSGA-II.

regarde dans la zone où $f_2 > 0.45$, l'écart n'est plus aussi facilement quantifiable. Ceci est le signe que la densité de la sortie f_1 pour les solutions Pareto-optimales a un support plus ou moins grand selon la position du point dans l'espace des objectifs.

Pour démontrer que le support de la distribution de la sortie $f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)$ peut grandement varier selon le point $\boldsymbol{x}$ où l'on se trouve, nous avons calculé la distribution de sortie en utilisant un échantillon Monte-Carlo de taille 10^4 autour de deux points du front de Pareto pire cas de la figure 7.9. Pour rappel, les lois des incertitudes avec lesquelles les entrées incertaines sont générées sont définies à la section 2.3.3. Les figures 7.13 et 7.14 illustrent ces deux distributions. Elles mettent en évidence deux points importants :

1. la diversité de densité de la sortie de f_1 que l'on obtient au niveau des points du PF pire cas. En effet, les deux densités en deux points différents du front ne se ressemblent pas. Celle de la figure 7.13 ressemble à une densité connue, que l'on pourrait par exemple approcher par une loi bêta. Celle de la figure 7.14 est plus atypique avec une queue de distribution assez lourde. En particulier, on peut remarquer que l'écart entre le quantile et le superquantile n'est pas le même dans ces deux densités. Ceci justifie donc l'utilisation d'un superquantile à la place du quantile.

2. le fait que la queue de la distribution puisse être lourde. La forme de la densité sur la figure 7.14 permet de remarquer que le pire cas se trouve dans une zone de cette dernière où les évènements sont peu probables. Notamment, on remarque que le quantile et le superquantile à 95% calculés sont très éloignés du pire cas, ce qui signifie que l'on se trouve dans un cadre où le pire cas retourne une solution trop conservative. La diversité de la forme des densités de sortie le long du PF ainsi que le large support de la distribution des solutions maximisant la durée de vie justifient l'utilisation d'une mesure probabiliste. Cela permet de trouver des solutions plus performantes en pondérant négativement la probabilité d'occurrence des évènements les plus extrêmes.

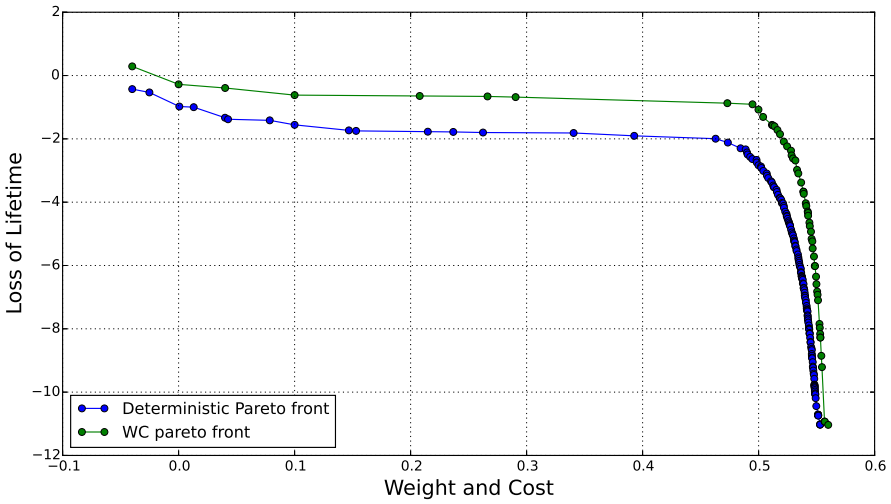


FIGURE 7.11 – Représentation dans l'espace des objectifs des points Pareto-optimaux pour la formulation déterministe et pour la formulation avec un pire cas pour quantifier les effets des incertitudes en entrée.

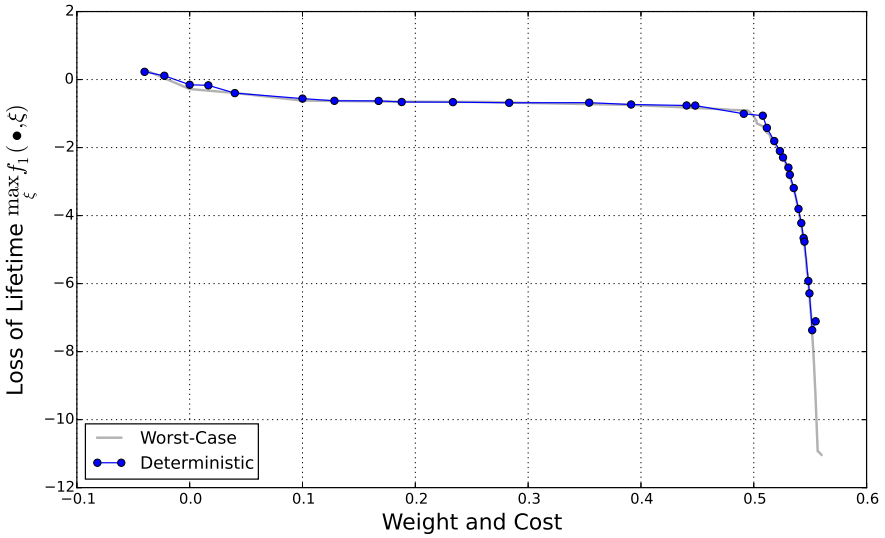


FIGURE 7.12 – Représentation du $\max_{\boldsymbol{\xi}} f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x},\boldsymbol{\xi}_x\right),\boldsymbol{\xi}_e\right)$ en fonction de f_2 des points Pareto-optimaux pour la formulation déterministe et pour la formulation avec un pire cas pour quantifier les effets des incertitudes en entrée.

7.2.3 Interprétation physique du résultat

Cette optimisation sous incertitudes est utilisée afin de proposer des solutions innovantes aux avionneurs. Il est donc important de donner une signification physique aux différentes solutions retournées par la méthode. Pour rappel, l'objectif de ce cas test était de comparer le résultat de l'équipement optimisé à un équipement de référence provenant d'un avion déjà existant. Les paramètres de contrôle $\boldsymbol{x}$ de l'optimisation traduisent diverses solutions de refroidissement, qui sont détaillées à la section 2.3.1.

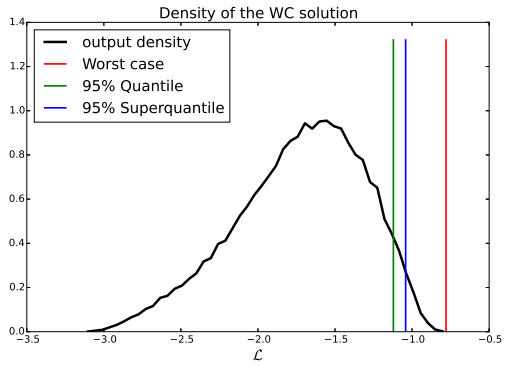


FIGURE 7.13 – Densité de la sortie autour du point du front de Pareto de la figure 7.9 de coordonnées $\max_\xi f_1 = -0.75$ et $f_2 = 0.472$

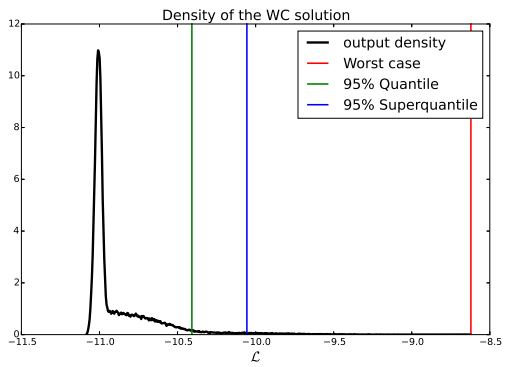


FIGURE 7.14 – Densité de la sortie autour du point du front de Pareto de la figure 7.9 de coordonnées $\max_\xi f_1 = -8.6$ et $f_2 = 0.554$

La figure 7.15 positionne l'équipement de référence par rapport au front de Pareto retourné par notre méthode. On remarque d'abord que l'équipement de référence n'est pas Pareto-optimal. Ensuite, il existe un large éventail de solutions que l'on peut diviser en trois groupes principaux (représentés sur la figure 7.15) :

1. Les solutions en orange dans la figure 7.15 qui sont des solutions aux caractéristiques thermiques proches de celles de l'équipement de référence. L'amélioration de la durée de vie obtenue se justifie par l'amélioration de la ventilation ou d'un traitement surfacique sur l'extérieur de l'équipement afin d'augmenter les échanges radiatifs avec l'environnement extérieur.
2. Les solutions en marron au milieu augmentent énormément la masse et le coût sans diminuer la durée de vie. Elles sont donc inutiles malgré l'utilisation de solutions de refroidissement plus innovantes telles que des caloducs (ou *heat pipes* en anglais). Ceci illustre un résultat contre-intuitif : selon les deux critères que l'on regarde, il n'est pas ou peu intéressant de changer un radiateur par des caloducs de basse ou moyenne qualité sur la paroi où se trouve le composant critique.
3. Les solutions innovantes augmentent la durée de vie de manière conséquente mais elles nécessitent un effort important au niveau du coût et de la masse pour l'avionneur. Ces solutions-là proposent une **rupture technologique**. En effet, elles utilisent des solutions de refroidissement peu utilisées en aéronautique : de nombreux caloducs de haute qualité se substituant à la ventilation et au traitement de surface sur les autres parois de l'équipement.

Il reste donc à présent à l'architecte de décider la solution qui lui convient le mieux : soit il décide de ne pas allouer plus de masse et de coût à cet équipement mais il n'améliore pas la durée de vie de son équipement. Soit il décide de faire un effort financier pour cet équipement pour allonger sa durée de vie.

Par ailleurs, la densité de la durée de vie pour une solution se situant dans la zone de rupture technologique montre que les incertitudes ont davantage d'effet sur ces dernières que sur des solutions comparables à l'équipement actuel. En particulier, nous avons mis en évidence le fait que la queue de la densité était plus lourde pour les solutions les plus innovantes. Nous pouvons donc espérer trouver de nouvelles solutions innovantes

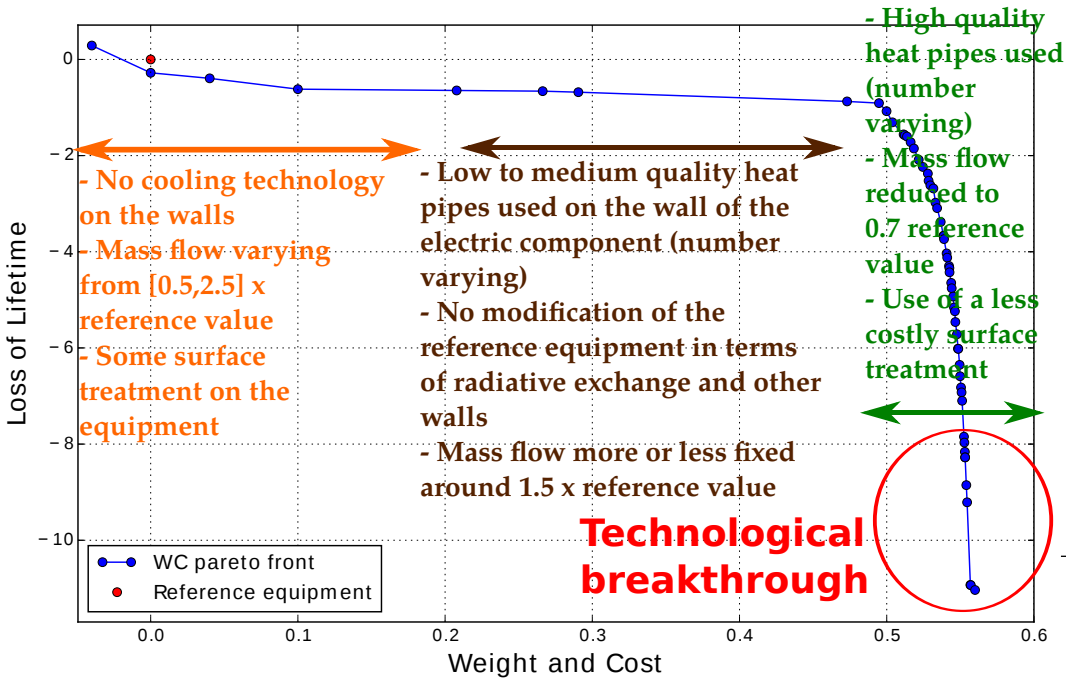


FIGURE 7.15 – Représentation dans l'espace des objectifs des points Pareto-optimaux pour la formulation pire cas. Comparaison avec les performances de l'équipement de référence et interprétation des solutions obtenues.

en modifiant la mesure de risque pire cas par une mesure probabiliste, cette dernière permettant de négliger les évènements trop peu probables.

Enfin, il est intéressant de remarquer que la prise en compte des incertitudes permet de découvrir de nouvelles solutions innovantes. En effet, la figure 7.12 montre que les solutions proposant une rupture technologique n'étaient pas présentes dans le front déterministe. Or, les ingénieurs sont particulièrement intéressés par ces solutions là. L'optimisation multi-objectif pire cas a donc permis de mieux répondre à leur besoin.

7.3 Conclusions sur l'optimisation pire cas par MOEGO NSGA-II couplé à EGO

Dans cette partie nous avons proposé un algorithme de résolution de problèmes multi-objectifs avec un objectif incertain. Pour cela :

— Nous avons prouvé la continuité de la fonction $g(\boldsymbol{x}) = \max_{\boldsymbol{\xi} \in \mathcal{D}_\xi} f(\boldsymbol{x}, \boldsymbol{\xi})$ au théorème 7.1 afin de pouvoir appliquer l'algorithme MOEGO NSGA-II développé dans la section 6.4.

— Nous avons choisi un algorithme mono-objectif permettant d'estimer cette fonction g en chaque point de contrôle $\boldsymbol{x}$ puis couplé ce dernier avec l'algorithme multi-objectif utilisé. Le couplage est quant à lui détaillé à l'algorithme 7.1. Nous avons également montré l'intérêt de cette méthode sur le cas analytique.

— Nous avons finalement appliqué cette méthodologie sur un cas analytique et un cas industriel sur lesquels nous avons montré l'efficience de cette stratégie. Pour un

temps de restitution de l'ordre de 1000 appels à la fonction avec entrées incertaines f_1, nous obtenons un front de Pareto incertain d'une qualité satisfaisante. Pour rappel, ces cas tests ont été utilisés avec quatre variables de contrôle et huit variables incertaines, dont les quatre de contrôle.

— Enfin, nous avons montré l'utilité sur le cas industriel et sur le cas analytique de la prise en compte des incertitudes lorsqu'elles existent. L'utilisation d'une mesure de risque de type pire cas présente l'avantage de trouver des solutions qui n'étaient pas présentes dans le front du problème déterministe et qui de plus ont l'avantage d'être robustes à la présence d'incertitudes. Elles sont optimales et dimensionnantes dans le sens où l'on suppose que leur durée de vie est la pire possible quelle que soit la valeur prise par les incertitudes sur les paramètres du problème.

De plus, dans le cas où les deux objectifs seraient incertains et nécessiteraient le calcul d'un pire cas, la méthode proposée ici serait tout à fait utilisable. En effet, il suffirait juste d'appliquer l'algorithme de réutilisation des appels aux deux fonctions objectif séparément, à l'aide de deux ensembles Ξ distincts.

Toutefois, ce chapitre a également fait ressortir des limites ou des besoins :

— La première limite est que le pire cas peut être une mesure de risque trop conservative. En effet, la probabilité d'occurrence de ce dernier est relativement réduite. La figure 7.14 montre qu'en utilisant un quantile ou un superquantile, on se donnerait la possibilité de trouver d'autres solutions robustes et plus performantes.

— La seconde concerne la réutilisation des points. L'efficacité de ce couplage dépend de l'amplitude des incertitudes sur les paramètres de contrôle, c'est-à-dire la taille de l'ensemble $\mathcal{D}_{\xi_x}$. En effet, si ce dernier est de trop petite taille, le nombre de points réutilisables, c'est-à-dire le cardinal de Ξ_x, est réduit. Néanmoins, cela simplifie aussi le calcul de la mesure de robustesse : si les incertitudes ont une faible amplitude, ceci signifie que l'on propage les incertitudes sur une zone des entrées de la fonction déterministe très localisée. Si cette fonction est suffisamment régulière, cela implique que plus le domaine $\mathcal{D}_{\xi_x}$ est localisé, moins les variations de la fonction dans cette zone sont importantes, simplifiant ainsi la propagation d'incertitudes.
Également, cette réutilisation des points va perdre de son efficacité lorsque la dimension des variables de contrôle va augmenter, à cause du fléau de la dimension. Plus précisément, la distance entre les points de contrôle $\boldsymbol{x}$ augmente avec la dimension N_x, ce qui tend à réduire le nombre de points réutilisables.

Chapitre 8

Optimisation Multi-objectif Probabiliste

Sommaire

Le chapitre précédent a montré que le pire cas pouvait être une mesure trop conservative. Notamment dans la section 7.2, le problème industriel illustre ce défaut puisque la densité de la sortie autour des solutions les plus innovantes a une queue particulièrement lourde. Or, ceci met en évidence le fait que le pire cas correspond à un évènement très improbable. L'utilisation d'une mesure probabiliste à la place du pire cas permet de prendre en compte la probabilité d'occurrence des évènements de la queue de distribution de la sortie. De cette manière, ces derniers peuvent être pondérés par leur probabilité d'occurrence, ce que ne permet pas de réaliser le pire cas. Le but de cette partie est donc d'utiliser l'une des mesures de risque ρ introduites au chapitre 4 à la place du pire cas et

de résoudre le problème suivant :

$$\begin{cases} \min\limits_{x} \rho_{\xi} \left[f_1(\boldsymbol{x}, \boldsymbol{\xi}_x, \boldsymbol{\xi}_e) \right] \\ \min\limits_{x} f_2(\boldsymbol{x}) \end{cases} \tag{8.1}$$

Nous devons d'abord choisir la mesure de risque parmi celles présentées dans le chapitre 4 la plus adaptée à notre problème. Cela est réalisé à la section 8.1. Pour rappel, les deux mesures candidates sont le quantile et le superquantile. La seconde présente l'avantage de prendre en compte les évènements se situant entre le quantile et le pire cas. Par ailleurs, l'algorithme d'optimisation sous incertitudes doit être applicable même dans le cas où les fonctions coût ont des temps d'exécution importants. Il est donc nécessaire de disposer d'un estimateur performant et parcimonieux de ces mesures de risque. Puisqu'il en existe peu dans la littérature pour le superquantile, la section 8.2 détaille un algorithme d'estimation de superquantile basé sur l'algorithme d'entropie croisée, initialement développé pour l'estimation de quantile. Afin de réduire le nombre d'appels à la fonction de référence, la méthode proposée peut également fonctionner avec un modèle de krigeage de la fonction de référence. Enfin, l'algorithme MOO probabiliste ainsi construit est appliqué à un cas test analytique en section 8.3 puis au cas industriel 8.4.

8.1 Justification du choix du superquantile

Dans la partie précédente, nous avons montré que les solutions qui traduisent une rupture technologique avaient une distribution particulière. En effet, nous avons vu que le support de la distribution de sortie de ces solutions était grand et que le pire cas avait une faible probabilité d'apparaître. C'est la raison pour laquelle il peut être considéré comme trop conservatif.

Pour pondérer négativement les évènements trop improbables, il est possible d'introduire des mesures de risque probabilistes telles que l'α-quantile ou l'α-superquantile. Ces mesures ont été étudiées au chapitre 4. La première consiste à négliger les $\alpha\%$ d'évènements trop extrêmes. La seconde, l'α-superquantile, peut être vue comme un complément à la première puisqu'elle consiste à moyenner l'effet des évènements se situant entre le quantile et le pire cas. La figure 4.2 illustre la signification de ces mesures sur une variable aléatoire de densité connue.

De ce fait, la mesure de superquantile peut être vue comme plus informative que le quantile. En effet, le quantile dépend de la queue de la distribution mais elle n'apporte pas d'informations sur la valeur des évènements plus extrêmes encore. En effet, le maximum peut se situer très loin du quantile et deux queues de distributions très différentes peuvent avoir une même valeur d'α-quantile. À l'inverse, le superquantile, en pondérant la queue de distribution entière, plaît davantage aux ingénieurs puisqu'il permet d'avoir des informations supplémentaires sur la forme de la queue de distribution. Puisqu'elle nécessite la connaissance du quantile afin d'être calculée, la comparaison des deux valeurs permet de détecter si la queue de distribution est lourde ou non, et si le maximum du support est éloigné ou non.

Pour répondre aux besoins des ingénieurs, nous avons donc décidé d'utiliser le superquantile comme mesure de risque appliquée au problème d'optimisation de l'équation (8.1). Pour le choix de la valeur du paramètre α, nous pouvons choisir un α non trop proche de 1 pour deux raisons. D'abord, nous souhaitons trouver des solutions différentes

du problème pire cas. Ensuite, le superquantile apporte une information sur l'extrêmité de la queue de la distribution. Même si nous ne calculons plus le pire cas, ce dernier est pris en compte dans le superquantile, même en ne prenant pas une valeur de α trop proche de 1. De ce fait, une valeur de $1-\alpha$ de l'ordre de 10^{-2} est raisonnable. Dans les faits, cette valeur dépend du besoin des ingénieurs et de leur tolérance au risque. Pour l'application, nous fixons $\alpha = 95\%$.

Le dernier point à vérifier pour pouvoir appliquer l'algorithme MOEGO NSGA-II est la continuité de l'application $g_\rho(\boldsymbol{x}) = \rho_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right]$. En effet, l'algorithme multi-objectif est basé sur une approximation par modèle de substitution des fonctions objectif. Or, la construction de modèle de substitution, dont la forme analytique est continue, nécessite la continuité des fonctions à substituer. Nous vérifions donc ici la continuité du superquantile avant d'en proposer un estimateur efficace.

Preuve de la continuité du $\rho_\xi\left[f_1(\boldsymbol{x},\boldsymbol{\xi})\right]$

Le théorème suivant prouve la continuité d'une mesure de robustesse avec de bonnes propriétés dans le cas où le domaine de définition de $\boldsymbol{\xi}$ ne dépend pas de $\boldsymbol{x}$:

Théorème 8.1. *Si $f \in \mathcal{C}^0\left(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R}\right)$ avec $\mathcal{D}_x \subset \mathbb{R}^{N_x}$ et $\mathcal{D}_\xi \subset \mathbb{R}^{N_\xi}$, deux ensembles compacts. Si ρ_ξ vérifie la propriété de monotonie et d'équivariance en translation (définies au chapitre 4.1 pour les mesures cohérentes), alors la fonction g_ρ définie à l'équation* (8.2) *est continue pour tout $\boldsymbol{x} \in \mathcal{D}_x$:*

$$g_\rho(\boldsymbol{x}) = \rho_{\boldsymbol{\xi}}\left[f(\boldsymbol{x},\boldsymbol{\xi})\right] \tag{8.2}$$

Preuve.
Soit $\boldsymbol{x}^0 \in \mathcal{D}_x$. L'application $(\boldsymbol{x},\boldsymbol{\xi}) \longmapsto f\left(\boldsymbol{x},\boldsymbol{\xi}\right)$ est continue sur $\mathcal{D}_x \times \mathcal{D}_\xi$ qui est un compact. Par le théorème de Heine, la fonction f est donc uniformément continue sur cet ensemble. Nous avons donc :

$$\forall \varepsilon > 0,\ \exists \delta > 0, \forall \boldsymbol{\xi} \in \mathcal{D}_\xi,\ \forall \boldsymbol{x} \text{ tel que } \left\|\boldsymbol{x}-\boldsymbol{x}^0\right\| \leqslant \delta \Rightarrow \left|f(\boldsymbol{x},\boldsymbol{\xi}) - f(\boldsymbol{x}^0,\boldsymbol{\xi})\right| \leqslant \varepsilon$$

Donc $\forall \boldsymbol{\xi} \in \mathcal{D}_\xi$, $\forall \boldsymbol{x}$ tel que $\|\boldsymbol{x}-\boldsymbol{x}^0\| \leqslant \delta$, on a :

$$\begin{array}{rcl} f(\boldsymbol{x},\boldsymbol{\xi}) & = & f(\boldsymbol{x},\boldsymbol{\xi}) - f(\boldsymbol{x}^0,\boldsymbol{\xi}) + f(\boldsymbol{x}^0,\boldsymbol{\xi}) \\ & \leqslant & \varepsilon + f(\boldsymbol{x}^0,\boldsymbol{\xi}) \end{array}$$

À présent, en utilisant les hypothèses de monotonie et d'invariance en translation de la mesure de robustesse ρ_ξ, on a :

$$\begin{array}{rcll} \Rightarrow \quad \rho_\xi\left[f(\boldsymbol{x},\boldsymbol{\xi})\right] & \leqslant & \rho_\xi\left[\varepsilon + f(\boldsymbol{x}^0,\boldsymbol{\xi})\right] & \text{(monotonie)} \\ & \leqslant & \varepsilon + \rho_\xi\left[f(\boldsymbol{x}^0,\boldsymbol{\xi})\right] & \text{(équivariance en translation)} \\ \Leftrightarrow \quad g_\rho(\boldsymbol{x}) - g_\rho(\boldsymbol{x}^0) & \leqslant & \varepsilon & \end{array}$$

En répétant le raisonnement précédent en partant de $f(\boldsymbol{x}^0,\boldsymbol{\xi})$ à la place de $f(\boldsymbol{x},\boldsymbol{\xi})$:

$$\begin{array}{rcl} f(\boldsymbol{x}^0,\boldsymbol{\xi}) & = & f(\boldsymbol{x}^0,\boldsymbol{\xi}) - f(\boldsymbol{x},\boldsymbol{\xi}) + f(\boldsymbol{x},\boldsymbol{\xi}) \\ & \leqslant & \varepsilon + f(\boldsymbol{x},\boldsymbol{\xi}) \end{array}$$

À présent, en utilisant les hypothèses de monotonie et d'invariance en translation de la mesure de robustesse ρ_ξ, on a :

$$\begin{array}{rlcll} \Rightarrow & \rho_\xi\left[f(\boldsymbol{x}^0,\boldsymbol{\xi})\right] & \leqslant & \rho_\xi\left[\varepsilon + f(\boldsymbol{x},\boldsymbol{\xi})\right] & \text{(monotonie)} \\ & & \leqslant & \varepsilon + \rho_\xi\left[f(\boldsymbol{x},\boldsymbol{\xi})\right] & \text{(équivariance en translation)} \\ \Leftrightarrow & g_\rho(\boldsymbol{x}^0) - g_\rho(\boldsymbol{x}) & \leqslant & \varepsilon & \end{array}$$

Ce qui signifie que pour tout $\boldsymbol{x}^0 \in \mathcal{D}_x$:

$$\forall \varepsilon > 0,\ \exists \delta > 0,\ \forall \boldsymbol{x} \in \mathcal{D}_x \left[\left\| \boldsymbol{x} - \boldsymbol{x}^0 \right\| \leqslant \delta \Rightarrow \left| g_\rho(\boldsymbol{x}^0) - g_\rho(\boldsymbol{x}) \right| \leqslant \varepsilon \right]$$

Ce qui prouve la continuité $\forall \boldsymbol{x}^0 \in \mathcal{D}_x$.

□

En remarquant que les conditions de monotonie et d'équivariance en translation sont vérifiées par un α-quantile et un α-superquantile à α fixé, ce théorème montre la continuité de ces mesures de robustesse.

De plus, les deux remarques réalisées à la section 7.1.1 restent valables ici. En particulier dans le cas où le domaine de variation de $\boldsymbol{\xi}$ dépend de $\boldsymbol{x}$, la continuité reste vraie en utilisant la même fonction $\boldsymbol{\zeta}$ qu'au chapitre 7. Cette fonction $\boldsymbol{\zeta}$ est donnée à l'équation (7.3).

8.2 Développement d'un estimateur de superquantile parcimonieux

Dans cette partie nous nous intéressons à l'estimation du superquantile de la sortie d'une boîte noire déterministe coûteuse ϕ avec des entrées suivant une variable aléatoire $\boldsymbol{\xi}$, comme au chapitre 4. Ceci permet de résoudre le problème de l'équation (8.1), puisqu'en chaque point $\boldsymbol{x} \in \mathcal{D}_x$, nous pouvons considérer $\phi\left(\boldsymbol{\xi}\right) = f\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)$. La variable $\boldsymbol{x}$ étant la partie déterministe, on peut ici s'affranchir de ce paramètre dans les notations.

Afin de pouvoir mettre en place un algorithme d'optimisation appliqué à cette mesure de robustesse, il est impératif de construire un estimateur du superquantile qui nécessite peu d'appels à la fonction ϕ. L'usage de modèles de substitution enrichis suivant un critère adapté est de ce fait l'un des objectifs de cette partie.

L'état de l'art de la section 4.2 montre que de nombreuses techniques existent pour estimer de manière parcimonieuse et performante un quantile ou une probabilité de dépassement de seuil, même dans le cas d'évènements rares. En revanche, lorsque l'on s'intéresse à l'estimation d'un superquantile, la liste de ces méthodes est plus réduite.

On peut donc s'intéresser aux méthodes appliquées au quantile et en déduire leur applicabilité au superquantile.

1. Par méthode de Monte-Carlo classique : comme le montre l'équation (4.8), la variance asymptotique de l'estimateur Monte-Carlo dépend du nombre d'échantillon en $\frac{1}{\sqrt{N}(1-\alpha)}$ où α est proche de 1 et avec N le nombre d'échantillons Monte-Carlo générés. De ce fait, pour compenser la valeur de $\frac{1}{1-\alpha}$ qui est de l'ordre de 10^2, il faut que la taille d'échantillon N vaille au minimum 10^4. Or, puisque nous souhaitons l'estimer en chaque $\boldsymbol{x}$ au cours de l'optimisation multi-objectif, ceci n'est pas envisageable.

2. Par échantillonnage stratifié : l'échantillonnage stratifié consiste à décomposer la quantité à estimer en produit de probabilités conditionnelles. Or, dans le cas du superquantile, il n'existe pas de manière évidente de se ramener à ce cadre théorique. De plus, il n'existe pas à notre connaissance de travaux basés sur l'échantillonnage stratifié adaptés à l'estimation du superquantile.
3. Par échantillonnage préférentiel : le superquantile est estimé par échantillonnage préférentiel pour chaque $\boldsymbol{x}$ en résolvant le problème de la formule (4.19) :

$$\tilde{Q}_\alpha^{IS} = \min_{\gamma \in \mathbb{R}} \Psi^{N_{IS}}(\gamma)$$
$$\text{avec } \Psi^{N_{IS}}(\gamma) = \left[\gamma + \frac{1}{(1-\alpha)\, N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \geqslant \gamma} \left(\phi(\tilde{\boldsymbol{\xi}}^i) - \gamma \right) \right] \tag{8.3}$$

le théorème 4.2 prouve l'existence et la convergence d'un estimateur par échantillonnage préférentiel du superquantile. Toutefois, l'échantillonnage préférentiel requiert le choix d'une distribution biaisée avec laquelle générer les échantillons. De plus, ce choix est crucial puisqu'une distribution auxiliaire mal choisie peut rendre la méthode moins efficace encore que la méthode de Monte-Carlo classique. Dans la littérature, l'unique proposition existante pour choisir cette distribution biaisée pour le superquantile provient de l'étude de Hong [Hong *et al.*, 2014]. Son idée consiste à rechercher la loi biaisée qui minimise la variance de l'estimateur. Pour cela, il recherche cette loi biaisée par la technique de changement de mesure exponentielle aussi appelée *exponential twisting* en anglais et notamment détaillée dans [Morio et Balesdent, 2015, pp.57-60]. Cependant, elle n'est adaptée qu'au cas où la densité de $\phi(\boldsymbol{\xi})$ est connue ou si la fonction déterministe ϕ dépend linéairement des incertitudes $\boldsymbol{\xi}$ [Morio et Balesdent, 2015, p.60]. Elle n'est de ce fait pas adaptée à notre étude.

On remarque donc que des techniques appliquées au quantile, la seule véritablement prometteuse pour l'estimation du superquantile est l'échantillonnage préférentiel. Toutefois, la mise en place de cet estimateur nécessite de choisir une distribution biaisée permettant de diminuer la variance de l'erreur commise. Pour cela, on peut s'inspirer de techniques existantes dans le cas du quantile dont la plupart sont présentées dans la section 4.2. L'algorithme de l'entropie croisée, détaillé à l'algorithme 4.2, représente une alternative prometteuse. Il est de plus adapté à l'utilisation de modèles de substitution, ce qui permet de davantage réduire le nombre d'appels à la fonction de référence ϕ. En effet, il existe dans la littérature divers critères d'enrichissement de plan d'expériences adaptés à l'estimation de quantiles. Ces derniers sont présentés à la section 4.2.3. Nous proposons donc en section 8.2.2 une adaptation au superquantile des critères existants pour le quantile.

8.2.1 Modification de l'algorithme d'entropie croisée

L'algorithme d'entropie croisée est introduit dans l'étude de [Rubinstein et Kroese, 2004]. Nous détaillons son application au cas du quantile dans la section 4.2.2.3. Il repose principalement sur deux idées :

1. La recherche de la distribution biaisée sur une famille de loi h_λ paramétrique. Pour en calculer les paramètres $\boldsymbol{\lambda}$, l'étude [Rubinstein et Kroese, 2004] propose

de minimiser la divergence de Kullback-Leibler entre la distribution optimale $h_b^\star$ minimisant la variance de l'estimateur et la distribution biaisée h_λ. Dans le cas du quantile, la distribution optimale est donnée à l'équation (4.18). Elle est par exemple illustrée sur la figure 8.1. Pour adapter cela au cas du superquantile, il est donc nécessaire de déterminer cette distribution optimale.

2. L'approximation de l'α-quantile à l'aide de β-quantiles empiriques (c'est-à-dire sans leur appliquer le rapport de vraisemblance) calculés sur des échantillons intermédiaires. La figure 8.2 illustre la manière d'utiliser ces β-quantiles empiriques afin d'approcher la distribution biaisée comme à l'algorithme 4.2. Pour que l'approche par ces β-quantiles empiriques fonctionne, il est nécessaire que $\beta < \alpha$. Plus précisément, β, conjugué au nombre d'échantillons N_{CE} utilisés à chaque itération, contrôle le nombre de points utilisés pour recalculer les paramètres $\boldsymbol{\lambda}$.

Afin d'adapter cet algorithme à l'estimation du superquantile, nous devons d'abord déterminer la loi biaisée optimale. Ceci permet de modifier la formule du calcul des paramètres $\boldsymbol{\lambda}$. Ensuite, afin de trouver la manière la plus adaptée d'utiliser l'algorithme de base de l'entropie croisée avec cette nouvelle formule, nous identifions les sources d'erreur de la formulation de l'équation (8.3). Ceci permet d'en tirer l'algorithme d'entropie croisée pour le calcul de superquantile.

8.2.1.1 Adaptation du calcul des paramètres de la loi biaisée à l'estimation du superquantile

Afin de déterminer la formule adaptée au calcul des paramètres biaisés optimaux dans le cas du superquantile, nous reprenons l'idée utilisée pour le quantile à la section 4.2.2.3. Le théorème 4.2, provenant de l'étude de Hong [Hong *et al.*, 2014], a montré que l'erreur d'estimation sur la fonction Ψ permettant d'estimer le superquantile est de la forme suivante :

$$\sigma_{Q,IS} = \frac{\sqrt{\mathrm{Var}\left(L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)\geqslant q_\alpha}\left(\phi\left(\tilde{\boldsymbol{\xi}}^i\right) - q_\alpha\right)\right)}}{(1-\alpha)} \tag{8.4}$$

On peut remarquer que la variance de cette erreur dépend de la loi biaisée h_b, même si elle n'apparaît pas directement dans l'équation précédente. Elle est incluse dans le rapport de vraisemblance $L(\tilde{\boldsymbol{\xi}}^i) = \frac{h\left(\tilde{\boldsymbol{\xi}}^i\right)}{h_b\left(\tilde{\boldsymbol{\xi}}^i\right)}$ (d'après l'équation (4.12)). Il est donc possible de trouver la loi biaisée optimale $h_b^\star$ qui permet d'annuler le numérateur de $\sigma_{Q,IS}$ donné à l'équation (8.4) :

$$\begin{aligned} &\mathrm{Var}\left(L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)\geqslant q_\alpha}\left(\phi\left(\tilde{\boldsymbol{\xi}}^i\right) - q_\alpha\right)\right) = 0 \\ \Leftrightarrow \quad &\int_{\boldsymbol{\xi}\in\mathcal{D}_\xi}\left(\frac{h(\boldsymbol{\xi})}{h_b^\star(\boldsymbol{\xi})}\left(\phi\left(\boldsymbol{\xi}\right) - q_\alpha\right)\mathbf{1}_{\phi(\boldsymbol{\xi})\geqslant q_\alpha} - Q_\alpha\right)^2 h_b^\star(\boldsymbol{\xi})d\boldsymbol{\xi} = 0 \end{aligned}$$

Or, si l'intégrale d'une fonction positive est nulle, alors cette fonction est nulle presque sûrement. Ceci fournit une solution pour la densité optimale $h_b^\star$:

$$h_b^\star(\boldsymbol{\xi}) = \frac{\mathbf{1}_{\phi(\boldsymbol{\xi})\geqslant q_\alpha}\left(\phi\left(\boldsymbol{\xi}\right) - q_\alpha\right)}{Q_\alpha}h(\boldsymbol{\xi}) \tag{8.5}$$

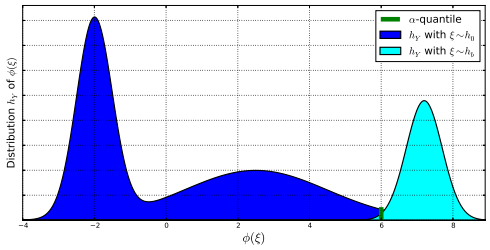
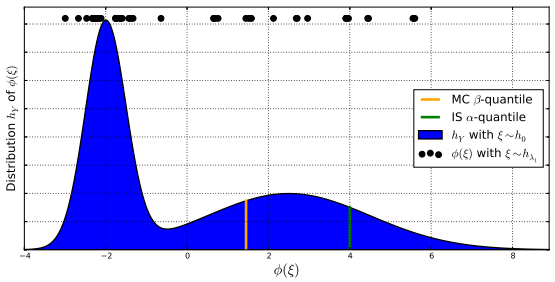
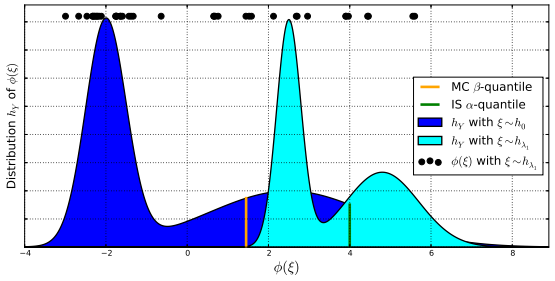


FIGURE 8.1 – Illustration de la distribution de sortie avec la loi biaisée optimale.

On échantillonne d'abord selon la vraie loi des entrées :



On en déduit le β-quantile empirique et l'α-quantile par échantillonnage préférentiel et l'on utilise le minimum des deux pour reconstruire une nouvelle loi biaisée :



On échantillonne ensuite selon la nouvelle loi biaisée et l'on continue le processus jusqu'à ce que l'α-quantile par échantillonnage préférentiel ne dépasse pas les β-quantiles empiriques

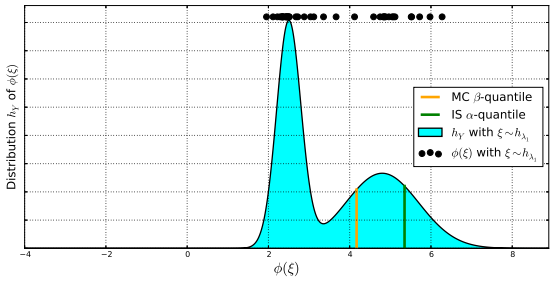


FIGURE 8.2 – Illustration des premières itérations de l'algorithme 4.2 d'entropie croisée permettant de s'approcher de la distribution optimale.

Comme dans le cas du quantile à l'équation (4.18), cette loi biaisée optimale est incalculable en l'état puisqu'elle dépend de deux paramètres *a priori* inconnus q_α et Q_α. C'est la raison pour laquelle nous devons utiliser une loi biaisée et des échantillons intermédiaires afin de l'approximer. Pour choisir les paramètres $\boldsymbol{\lambda}$ de la loi biaisée optimale $h_{\boldsymbol{\lambda}}$, nous minimisons la divergence de Kullback-Leibler [Joyce, 2011] entre ces deux mesures. Cette quantité permet de quantifier la dissimilarité entre deux distributions. Le problème

d'optimisation à résoudre s'écrit donc :

$$\begin{aligned} \boldsymbol{\lambda}^\star &= \underset{\boldsymbol{\lambda}}{\operatorname{argmin}}\ D(h_b^\star, h_{\boldsymbol{\lambda}}) \\ &= \underset{\lambda}{\operatorname{argmin}}\ \left[\int_{\mathbb{R}^{N_\xi}} h_b^\star(\boldsymbol{\chi}) \ln\left[h_b^\star(\boldsymbol{\chi})\right] d\boldsymbol{\chi} - \int_{\mathbb{R}^{N_\xi}} h_b^\star(\boldsymbol{\chi}) \ln\left[h_{\boldsymbol{\lambda}}(\boldsymbol{\chi})\right] d\boldsymbol{\chi}\right] \end{aligned} \tag{8.6}$$

Les termes ne dépendant pas de $\boldsymbol{\lambda}$ dans l'équation (4.21) n'influent pas sur les solutions du problème d'optimisation à résoudre, on peut donc oublier ces termes constants par rapport à $\boldsymbol{\lambda}$, ce qui donne en développant $h_b^\star$ d'après l'équation (8.5) :

$$\boldsymbol{\lambda}^\star = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}} \int_{\mathbb{R}^{N_\xi}} \frac{\mathbf{1}_{\phi(\boldsymbol{\chi})\geqslant q_\alpha} \left(\phi\left(\boldsymbol{\chi}\right) - q_\alpha\right) \ln\left[h_{\boldsymbol{\lambda}}(\boldsymbol{\chi})\right]}{Q_\alpha} h(\boldsymbol{\chi}) d\boldsymbol{\chi}$$

Une fois encore en oubliant les constantes, comme la cible Q_α, et en se ramenant à une écriture probabiliste, on peut donc écrire :

$$\boldsymbol{\lambda}^\star = \underset{\lambda}{\operatorname{argmax}}\ \mathbb{E}\left(\mathbf{1}_{\phi(\boldsymbol{\xi})\geqslant q_\alpha} \left(\phi\left(\boldsymbol{\xi}\right) - q_\alpha\right) \ln(h_{\boldsymbol{\lambda}}(\boldsymbol{\xi}))\right) \tag{8.7}$$

En utilisant l'estimation par échantillonnage préférentiel, nous obtenons :

$$\boxed{\boldsymbol{\lambda}^\star = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}}\ \frac{1}{N_{CE}} \sum_{i=1}^{N_{CE}} \left(\phi\left(\tilde{\boldsymbol{\xi}}^i\right) - q_\alpha\right) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} L(\tilde{\boldsymbol{\xi}}^i) \ln(h_{\boldsymbol{\lambda}}(\tilde{\boldsymbol{\xi}}^i))} \tag{8.8}$$

L'équation (8.8) illustre le fait que, pour estimer le superquantile par entropie croisée, il est d'abord nécessaire d'estimer le quantile. Afin d'insister sur ce point, la section 8.2.1.2 permet de montrer que l'erreur d'estimation du superquantile par échantillonnage préférentiel est également majorée par l'erreur commise sur l'estimation du quantile.

Remarque sur l'équation (8.8) **:** en comparant les équations à résoudre dans le cas du quantile (équation (4.23)) et du superquantile (équation (8.8)), on remarque que l'unique différence entre les deux est l'ajout du terme $\left(\phi\left(\tilde{\boldsymbol{\xi}}^i\right) - q_\alpha\right)$ à la formule pour le superquantile. On peut donc réécrire cette équation d'une manière plus générale :

$$\boldsymbol{\lambda}^\star = \underset{\boldsymbol{\lambda}}{\operatorname{argmax}}\ \frac{1}{N_{CE}} \sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right) \ln(h_{\boldsymbol{\lambda}}(\tilde{\boldsymbol{\xi}}^i)) \tag{8.9}$$

La fonction $\Theta\left(\tilde{\boldsymbol{\xi}}^i\right)$ valant par exemple $\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i)>q_\alpha} L(\tilde{\boldsymbol{\xi}}^i)$ dans le cas du quantile.

Tout d'abord, ceci est cohérent avec l'interprétation du superquantile. En effet, il consiste à pondérer les évènements se situant au dessus du quantile, ce que traduit tout à fait l'ajout du terme $\left(\phi\left(\tilde{\boldsymbol{\xi}}^i\right) - q_\alpha\right)$ dans la formule de Θ pour le superquantile par rapport à celle adaptée au quantile. Ensuite, il est important de remarquer que cette fonction Θ ne dépend pas de $\boldsymbol{\lambda}$. Ainsi, ceci présente l'avantage de ne pas complexifier le calcul des paramètres optimaux dans le cas où la solution était analytique pour le quantile. En particulier, si $h_{\boldsymbol{\lambda}}$ suit une loi normale, la solution est analytique. Soient $\lambda_{j,1}$ la moyenne et

$\lambda_{j,2}$ l'écart type pour chaque composante $j \in \{1, \cdots, N_\xi\}$. Comme à la section 4.2.2.3, on a dans ce cas la solution suivante à l'équation (8.9) :

$$\begin{cases} \hat{\lambda}^\star_{j,1} &= \dfrac{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right) \tilde{\xi}^i_j}{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right)} \\ \hat{\lambda}^\star_{j,2} &= \dfrac{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right) (\tilde{\xi}^i_j - \lambda^\star_{j,1})^2}{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right)} \end{cases} \tag{8.10}$$

8.2.1.2 Contribution de l'erreur du quantile dans l'erreur totale pour l'estimation du superquantile par échantillonnage préférentiel

Pour comprendre comment utiliser l'algorithme d'entropie croisée, nous devons étudier l'erreur de l'estimateur du superquantile introduit par Rockafellar à l'équation (8.3). Pour évaluer l'erreur d'estimation du superquantile $Q_\alpha = \Psi(q_\alpha)$, il nous faut d'abord quantifier l'erreur commise en l'optimum $\gamma^\star$ qui correspond au quantile q_α et que l'on ne connaît que par son approximation $\tilde{\gamma}_{N_{IS}} = \tilde{q}^{IS}_\alpha$:

$$\Psi^{N_{IS}}(\tilde{q}^{IS}_\alpha) - \Psi(q_\alpha) = \underbrace{\Psi^{N_{IS}}(\tilde{q}^{IS}_\alpha) - \Psi^{N_{IS}}(q_\alpha)}_{\text{inconnu}} + \underbrace{\Psi^{N_{IS}}(q_\alpha) - \Psi(q_\alpha)}_{\text{estimé au théorème 4.2}} \tag{8.11}$$

Proposition 8.1. *Si* $\mathbb{E}\left[L(\tilde{\boldsymbol{\xi}}^1)^2\right] < \infty$, *alors :*

$$\left|\Psi^{N_{IS}}(\tilde{q}^{IS}_\alpha) - \Psi^{N_{IS}}(q_\alpha)\right| \leqslant \left|W^{N_{IS}}\right|$$

où $W^{N_{IS}} \underset{N_{IS}\to\infty}{\overset{\mathcal{L}}{\to}} \sigma_{q,IS}\mathcal{N}(0,1)$ *et* $\sigma_{q,IS}$ *est défini au théorème 4.1 à l'équation* (4.15).

Preuve.
Posons $\Sigma^{N_{IS}} = \Psi^{N_{IS}}(\tilde{q}^{IS}_\alpha) - \Psi^{N_{IS}}(q_\alpha)$. On a :

$$\Sigma^{N_{IS}} = \frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} \left\{ \left[\phi(\tilde{\boldsymbol{\xi}}^j) - \tilde{q}^{IS}_\alpha\right] \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \geqslant \tilde{q}^{IS}_\alpha} - \left[\phi(\tilde{\boldsymbol{\xi}}^j) - q_\alpha\right] \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \geqslant q_\alpha} \right\} L(\tilde{\boldsymbol{\xi}}^j)$$

En séparant les indicatrices en évènements disjoints, on obtient :

$$\Sigma^{N_{IS}} = \frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j) \left\{ \begin{array}{l} (q_\alpha - \tilde{q}^{IS}_\alpha) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^i) \geqslant \max(q_\alpha, \tilde{q}^{IS}_\alpha)} \\ + \left(\phi(\tilde{\boldsymbol{\xi}}^j) - \tilde{q}^{IS}_\alpha\right) \mathbf{1}_{\max(q_\alpha, \tilde{q}^{IS}_\alpha) \geqslant \phi(\tilde{\boldsymbol{\xi}}^i) \geqslant \tilde{q}^{IS}_\alpha} \\ + \left(\phi(\tilde{\boldsymbol{\xi}}^j) - q_\alpha\right) \mathbf{1}_{\max(q_\alpha, \tilde{q}^{IS}_\alpha) \geqslant \phi(\tilde{\boldsymbol{\xi}}^i) \geqslant q_\alpha} \end{array} \right\}$$

En majorant les termes où la fonction ϕ se trouve en dehors de l'indicatrice, on a :

$$\left|\Sigma^{N_{IS}}\right| \leqslant \frac{\left|\tilde{q}^{IS}_\alpha - q_\alpha\right|}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant \min(q_\alpha, \tilde{q}^{IS}_\alpha)} \tag{8.12}$$

Le lemme suivant permet de déterminer la limite du majorant dans l'équation (8.12) quand N_{IS} tend vers l'infini.

Lemme 8.1. *Si* $\mathbb{E}\left[L(\tilde{\boldsymbol{\xi}}^1)^2\right] < \infty$, *alors :*

$$v^{N_{IS}} \xrightarrow[N_{IS}\to\infty]{\mathbb{P}} 1$$

avec $v^{N_{IS}} = \frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \min(q_\alpha, \tilde{q}_\alpha^{IS})}$

Preuve.
Regardons l'évènement sur lequel porte l'indicatrice de la formule de $v^{N_{IS}}$. On a :

$$\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \min(q_\alpha, \tilde{q}_\alpha^{IS})} = \mathbf{1}_{\tilde{q}_\alpha^{IS} > \phi(\tilde{\boldsymbol{\xi}}^j)\geqslant q_\alpha} + \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \tilde{q}_\alpha^{IS} > q_\alpha} + \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant q_\alpha \geqslant \tilde{q}_\alpha^{IS}} + \mathbf{1}_{q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \tilde{q}_\alpha^{N_{IS}}}$$

Posons :

$$\left\{\begin{array}{rcl} A & := & \left\{\tilde{q}_\alpha^{IS} > \phi(\tilde{\boldsymbol{\xi}}^j) \geqslant q_\alpha\right\} \\ B & := & \left\{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant \tilde{q}_\alpha^{IS} > q_\alpha\right\} \\ C & := & \left\{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant q_\alpha \geqslant \tilde{q}_\alpha^{IS}\right\} \\ D & := & \left\{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant q_\alpha\right\} \end{array}\right.$$

Puisque $D = A \cup B \cup C$ et que $A \cap B = \emptyset$, $A \cap C = \emptyset$ et $B \cap C = \emptyset$, on a donc :

$$\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \min(q_\alpha, \tilde{q}_\alpha^{IS})} = \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant q_\alpha} + \mathbf{1}_{q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \tilde{q}_\alpha^{N_{IS}}}$$

On en déduit que :

$$v^{N_{IS}} = \underbrace{\frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant q_\alpha}}_{v_1^{N_{IS}}} + \underbrace{\frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\mathbf{1}_{q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \tilde{q}_\alpha^{N_{IS}}}}_{v_2^{N_{IS}}}$$

Pour le terme de gauche, $v_1^{N_{IS}}$, nous pouvons appliquer la loi des grands nombres [Saporta, 2006, p.277] sur la variable $L(\tilde{\boldsymbol{\xi}}^1)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant q_\alpha}$. En effet, cette quantité est de variance finie (par hypothèse de l'algorithme) et est de moyenne E avec, par définition de q_α :

$$E = \mathbb{E}\left[L(\tilde{\boldsymbol{\xi}}^j)\mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j)\geqslant q_\alpha}\right] = \mathbb{P}\left(\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant q_\alpha\right) = 1 - \alpha$$

Ainsi, $v_1^{N_{IS}} \xrightarrow[N_{IS}\infty]{\mathbb{P}} 1$

Il reste donc à étudier le terme $v_2^{N_{IS}} = \frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\mathbf{1}_{q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^j)\geqslant \tilde{q}_\alpha^{N_{IS}}}$. On a $v_2^{N_{IS}} \geqslant 0$, on peut donc appliquer l'inégalité de Markov. Ceci nous donne :

$$\forall t > 0, \quad \mathbb{P}\left[v_2^{N_{IS}} > t\right] \leqslant \frac{\mathbb{E}\left(v_2^{N_{IS}}\right)}{t} \tag{8.13}$$

Or

$$\begin{array}{rcl} \mathbb{E}\left(v_2^{N_{IS}}\right) & = & \mathbb{E}\left[L(\tilde{\boldsymbol{\xi}}^1)\mathbf{1}_{q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^1)\geqslant \tilde{q}_\alpha^{N_{IS}}}\right] \\ & \leqslant & \sqrt{\left(\mathbb{E}\left(L(\tilde{\boldsymbol{\xi}}^1)^2\right)\mathbb{E}\left(\mathbf{1}_{q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^1)\geqslant \tilde{q}_\alpha^{N_{IS}}}\right)\right)} \text{par l'inégalité de Cauchy-Schwartz} \\ & \leqslant & C\mathbb{P}\left(q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^1) \geqslant \tilde{q}_\alpha^{N_{IS}}\right) \end{array}$$

Le terme $\mathbb{E}\left(L(\tilde{\boldsymbol{\xi}}^1)^2\right)$ est borné par une constante, par hypothèse. Le couple $\left(\tilde{\boldsymbol{\xi}}_1, \tilde{q}_\alpha^{N_{IS}}\right)$ appartient au fermé $F = \{(x,y) \in \mathbb{R}^2 \ : \ q_\alpha \geqslant x \geqslant y\}$. De plus, $\left(\tilde{\boldsymbol{\xi}}_1, \tilde{q}_\alpha^{N_{IS}}\right) \xrightarrow[N_{IS}\to\infty]{\mathbb{P}} \left(\tilde{\boldsymbol{\xi}}_1, q_\alpha\right)$ d'après le théorème 4.1. On peut donc appliquer le théorème porte-manteau [Billingsley, 2008] ce qui nous donne que :

$$\lim_{N_{IS}\to\infty} \mathbb{P}\left(q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^1) \geqslant \tilde{q}_\alpha^{N_{IS}}\right) \leqslant \mathbb{P}\left(q_\alpha \geqslant \phi(\tilde{\boldsymbol{\xi}}^1) \geqslant q_\alpha\right) = 0$$

En remontant à l'équation (8.13), on obtient donc que

$$\forall t > 0, \ \lim_{N_{IS}\to\infty} \mathbb{P}\left[v_2^{N_{IS}} > t\right] = 0$$

Ce qui est la définition de la convergence en probabilité de $v_2^{N_{IS}}$ vers 0. On obtient donc que le terme $v^{N_{IS}}$ converge en probabilité (et donc en loi) vers 1.

□

Le lemme précédent montre que la composante $v^{N_{IS}} = \frac{1}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant \min(q_\alpha, \tilde{q}_\alpha^{IS})}$ tend vers 1 en probabilité lorsque $N_{IS} \to \infty$. De plus, le théorème 4.1 montre la convergence en loi de $\sqrt{N_{IS}}\left(\tilde{q}_\alpha^{IS} - q_\alpha\right)$ vers $\sigma_{q,IS}\mathcal{N}(0,1)$ où $\sigma_{q,IS}$ est définie à l'équation (4.15).

Donc grâce au théorème de Slutsky [Grimmett et Stirzaker, 2001, p.318], on obtient le résultat suivant :

$$\begin{aligned}\sqrt{N_{IS}} \frac{\tilde{q}_\alpha^{IS} - q_\alpha}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant \min(q_\alpha, \tilde{q}_\alpha^{IS})} &= \sqrt{N_{IS}}\left(\tilde{q}_\alpha^{IS} - q_\alpha\right) \times v^{N_{IS}} \\ &\xrightarrow[N_{IS}\to\infty]{\mathcal{L}} \sigma_{q,IS}\mathcal{N}(0,1) \times 1 = \sigma_{q,IS}\mathcal{N}(0,1)\end{aligned}$$

En posant $W^{N_{IS}} = \sqrt{N_{IS}} \frac{\tilde{q}_\alpha^{IS} - q_\alpha}{N_{IS}(1-\alpha)} \sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j) \mathbf{1}_{\phi(\tilde{\boldsymbol{\xi}}^j) \geqslant \min(q_\alpha, \tilde{q}_\alpha^{IS})}$, et puisque $\sqrt{N_{IS}} \geqslant 1$, l'équation (8.12) permet donc d'obtenir le résultat annoncé.

□

Remarque : en repartant de l'équation (8.11) et grâce à la proposition 8.1 et au Théorème 4.2, nous avons mis en évidence que l'erreur commise sur le superquantile par l'équation (4.19) est asymptotiquement majorée par l'erreur commise sur le quantile et par celle commise sur l'approximation de Ψ. Ceci démontre ainsi formellement un résultat qui paraissait intuitif.

8.2.1.3 Algorithme d'entropie croisée pour le superquantile

La proposition 8.1 ainsi que l'équation (8.8) mettent donc en évidence qu'avant d'estimer l'α-superquantile par l'algorithme d'entropie croisée, il est d'abord utile d'approcher l'α-quantile. Plus précisément, ceci signifie que nous devons d'abord utiliser l'algorithme 4.2 avant d'estimer le superquantile à l'aide d'un échantillon généré par la loi biaisée calculée grâce à l'équation (8.8). L'algorithme 8.1 résume cette idée.

Algorithme 8.1 Algorithme de l'entropie croisée pour l'estimation d'un α-superquantile

Input : la densité jointe des entrées h, la fonction $\phi(\cdot)$, le nombre d'échantillons N_{CE} par itération, la valeur des quantiles intermédiaires $\beta \in [0,1]$

Output : le superquantile $\tilde{Q}_\alpha^{IS}$

· Exécuter l'algorithme 4.2 en récupérant $\tilde{q}_\alpha^{IS}$ ainsi que le dernier échantillon $\tilde{\boldsymbol{\xi}}^1, \cdots, \tilde{\boldsymbol{\xi}}^{N_{CE}}$ et sa loi h_{λ_q};

· Optimiser les paramètres de la densité biaisée $\boldsymbol{\lambda}^\star$ selon l'équation (8.8) avec $\tilde{q}_\alpha^{IS}$ à la place de q_α;

· Générer les échantillons i.i.d $\tilde{\boldsymbol{\xi}}^1, \cdots, \tilde{\boldsymbol{\xi}}^{N_{CE}}$ avec la densité $h_{\boldsymbol{\lambda}^\star}$;

· Estimer le superquantile $\tilde{Q}_\alpha^{IS}$ avec le dernier échantillon généré par l'équation (8.3);

Remarque : Pour l'estimation de la loi biaisée, nous avons déjà précisé que dans le cas de la loi normale, le résultat de la résolution de (8.9) est analytique. Toutefois, le cas de la loi normale peut être considéré comme trop restrictif.

Une loi assez répandue dans le but d'approximer des lois de probabilité bornées est la loi bêta donnée en annexe A. Elle présente l'avantage d'être très flexible, comme l'illustre la figure A.2. Pour calculer les paramètres optimaux $\lambda^\star = (a^\star, b^\star)$, il suffit de calculer la valeur de ces paramètres qui annulent le gradient de la divergence de Kullback-Leibler. On a donc :

$$\frac{1}{N_{CE}} \sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right) \ \nabla \ln(h_\lambda(\tilde{\boldsymbol{\xi}}^i)) = 0 \tag{8.14}$$

Or dans le cas d'une loi Bêta, on a :

$$\frac{\partial}{\partial a} \ln(h_{a,b}(t)) = \frac{\partial h_{a,b}(t)}{\partial a} \frac{1}{h_{a,b}(t)}$$

De plus,

$$\frac{\partial h_{a,b}(t)}{\partial a} = \ln(t) h_{a,b}(t) - \frac{\partial B(a,b)}{\partial a} \frac{h_{a,b}(t)}{B(a,b)} = \Big[\ln(t) - (\psi(a) + \psi(a+b)) \Big] h_{a,b}(t)$$

où B est la fonction Bêta et ψ la fonction digamma. De la même manière pour la dérivée par rapport à b :

$$\frac{\partial h_{a,b}(t)}{\partial b} = \Big[\ln(1-t) - (\psi(b) + \psi(a+b)) \Big] h_{a,b}(t)$$

Ce qui nous donne pour la composante $j \in \{1, \cdots, N_\xi\}$ si l'on se ramène au gradient de la divergence à l'équation (8.14) et en considérant la symétrie de la fonction Bêta :

$$\begin{cases} \psi(a_j) + \psi(a_j + b_j) = \dfrac{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right) \ln(\tilde{\xi}_j^i)}{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right)} \\ \psi(b_j) + \psi(a_j + b_j) = \dfrac{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right) \ln(1 - \tilde{\xi}_j^i)}{\sum_{i=1}^{N_{CE}} \Theta\left(\tilde{\boldsymbol{\xi}}^i\right)} \end{cases} \tag{8.15}$$

Cette équation (8.15) peut être résolue par une minimisation du type moindres carrées. Ceci revient à rechercher les paramètres a_i et b_i qui annulent la différence entre les termes se situant de chaque côté de l'égalité de cette équation. Toutefois, cette résolution n'est pas plus aisée que la résolution directe de l'équation (8.9). En dehors de certaines familles paramétriques particulières de loi (telles que la loi normale ou la loi triangulaire [Morio et Balesdent, 2015, pp.61-62]), la résolution de l'équation (8.9) n'est pas aisée. Dans l'application de la section 8.2.3, nous nous concentrons sur la famille des lois normales (tronquées ou non).

8.2.2 Modification de l'enrichissement du modèle de krigeage

La section 4.2.3 détaille tous les critères d'enrichissement permettant d'améliorer la précision dans l'estimation d'un quantile à l'aide de la prédiction par krigeage de la fonction ϕ. Dans cette partie, nous partons de l'un de ces critères afin de l'adapter au calcul du superquantile.

Puisque le calcul du superquantile doit être fait en chaque point $\boldsymbol{x}$, il est indispensable que le critère d'enrichissement utilisé ne soit pas trop gourmand en temps de calcul. C'est la raison pour laquelle nous excluons ici les méthodes du type SUR, détaillées à l'équation (4.30). Cette technique permet d'enrichir le plan d'expériences avec le point qui diminue le plus la variance de l'estimation de la mesure de robustesse par le modèle de krigeage utilisé. Or, pour chaque rajout de points, ceci nécessite la maximisation d'un critère qui peut être très coûteux à estimer. De plus, puisque nous calculons le quantile ou le superquantile à l'aide d'un échantillon Monte-Carlo fixé, la technique inspirée de l'étude de Balesdent [Balesdent *et al.*, 2013] répond davantage à la problématique. En effet, elle consiste à rajouter les points qui diminuent le plus l'incertitude des points de l'échantillon Monte-Carlo dont la position par rapport au quantile est incertaine à cause de l'erreur d'estimation du krigeage. Ce point de vue est détaillé en section 4.2.3. Bien qu'également coûteuse, l'approche présente l'avantage de répondre exactement à notre objectif en diminuant uniquement l'incertitude des points de l'échantillon Monte-Carlo nécessaires au calcul de la mesure de risque étudiée. En d'autres termes, le critère SUR présente ici le désavantage d'être trop coûteux et trop *idéalisé*.

Dans le cas du quantile, puisque la fonction ϕ est uniquement présente dans l'estimation du quantile et de la loi biaisée par l'intermédiaire d'une indicatrice, il est inutile de connaître la fonction ϕ précisément. Il faut uniquement connaître les zones de l'espace des entrées où cette fonction est au dessus du seuil courant ou non. Pour cela, il est donc possible de se limiter à l'utilisation d'un modèle de substitution de type classifieur.

Par ailleurs, l'unique différence entre le calcul du quantile et du superquantile est l'apparition du terme $(\phi(\boldsymbol{\xi}) - \gamma)$ dans l'estimation des paramètres optimaux de la loi biaisée. De ce fait, pour le superquantile, il ne suffit plus de savoir la position de $\phi(\boldsymbol{\xi})$ par rapport au seuil, mais il faut également connaître l'amplitude du dépassement de ce seuil. Dès lors, pour l'estimation de superquantile il est nécessaire de modifier la manière d'enrichir le plan d'expériences.

Plus précisément, il n'est en fait nécessaire de connaître la fonction qu'au-delà du quantile recherché. De ce fait, le critère proposé par Echard dans son étude [Echard *et al.*, 2013] et détaillé à l'équation (4.29) ne peut être applicable ici, car, par construction, il ne permet que de quantifier l'amélioration de la connaissance d'un seuil.

Le seul critère présenté à la section 4.2.3 qui peut être étendu au calcul du super-

quantile est donc celui utilisé dans l'étude de Balesdent [Balesdent *et al.*, 2013] et donné à l'équation (4.27). Pour cela, l'unique modification sur l'enrichissement proposé à la section 4.2.3 porte sur la définition de l'ensemble $U_{k,q}$ de l'équation (4.26). Pour rappel, $U_{k,q}$ correspond à l'ensemble des points Monte-Carlo dont le positionnement par rapport au quantile calculé sur $\hat{\phi}$ est incertain. L'incertitude dont il est question ici est celle définie par cette variance de la prédiction par krigeage. L'idée du critère de l'équation (4.27) est ensuite de diminuer la variance de krigeage moyenne des points de $U_{k,q}$, ce qui revient à choisir le point maximisant le critère IMSE (équation (3.25)) restreint à cet ensemble (voir l'équation (4.27)).

Pour adapter cette idée au superquantile, il suffit donc ici de modifier la définition de l'ensemble des points dont on souhaite diminuer l'erreur d'estimation. Plus précisément, il s'agit de tous les points qui sont potentiellement au dessus du quantile estimé. En d'autres termes :

$$U_{\alpha,Q} = \left\{ \boldsymbol{\xi} \in \left(\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^{N_{MC}} \right) \ : \ \tilde{q}_{\alpha_{\hat{\phi}(\bullet)-k\hat{\sigma}(\bullet)}} < \hat{\phi}(\boldsymbol{\xi}) + k\hat{\sigma}(\boldsymbol{\xi}) \right\} \tag{8.16}$$

Cet ensemble incertain est illustré sur la figure 8.3. On remarque dans cette équation

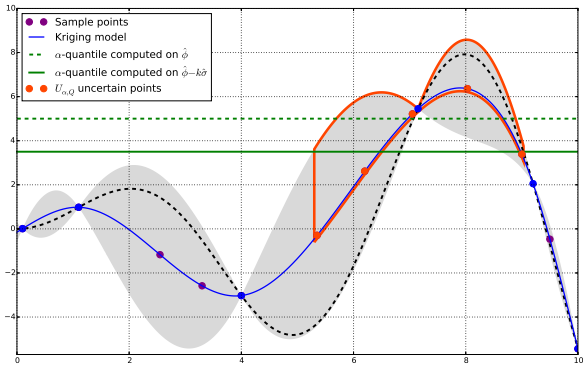


FIGURE 8.3 – Illustration de la zone incertaine $U_{\alpha,Q}$ à enrichir pour l'estimation d'un α-superquantile par krigeage

que le quantile estimé $\tilde{q}_{\alpha_{\hat{\phi}(\bullet)-k\hat{\sigma}}}$ est le quantile estimé sur la borne inférieure de la prédiction par krigeage, $\hat{\phi}(\bullet) - k\hat{\sigma}(\bullet)$. Ceci permet de prendre également en compte l'erreur potentiellement réalisée sur l'estimation du quantile lui-même. L'enrichissement du plan d'expériences de krigeage dans l'étape d'estimation du superquantile est résumé à l'algorithme 8.2. Ce dernier est donc appliqué à chaque étape de l'algorithme 8.1 où la valeur de ϕ doit être connue au-delà du superquantile. Ceci signifie que cet enrichissement est réalisé avant la résolution de l'équation (8.8) et avant l'estimation finale du superquantile.

8.2.3 Application de l'algorithme d'entropie croisée pour le superquantile

8.2.3.1 Cas tests utilisés

Pour valider l'algorithme d'estimation du superquantile, nous utilisons quatre fonctions de référence. Les lois de probabilités sur les entrées choisies ici sont des lois normales ou des lois normales tronquées et sont définies dans l'annexe A.

Algorithme 8.2 Enrichissement séquentiel du modèle de krigeage pour le calcul du superquantile

Input : Un *DOE* initial (généré par LHS par exemple)
un échantillon Monte-Carlo $\left(\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^{N_{MC}}\right)$
La taille de l'intervalle de confiance k (1.96 par défaut)
le niveau α de quantile recherché

Output : la valeur du quantile $\hat{Q}_\alpha$

· Construire $\hat{\phi}$ à partir du *DOE* (avec son estimation d'erreur $\hat{\sigma}$);
· Calculer l'α-quantile $\tilde{q}_{\alpha_{\hat{\phi}-k\hat{\sigma}}}$ à l'aide de la borne inférieure de krigeage sur cet échantillon;
· En déduire le domaine de confiance $U_{\alpha,Q}$ (défini à l'équation (8.16));
· En déduire également la valeur du superquantile sur la borne inférieure et supérieure du krigeage $Q_{\hat{\phi}-k\hat{\sigma}}$ et $Q_{\hat{\phi}+k\hat{\sigma}}$;

while card $(U_{\alpha,Q}) \neq 0$ *ou* $\frac{Q_{\hat{\phi}+k\hat{\sigma}} - Q_{\hat{\phi}-k\hat{\sigma}}}{Q_{\hat{\phi}-k\hat{\sigma}}} > \varepsilon$ **do**
· Choisir $\boldsymbol{\xi}^\star = \text{argmax}_{\boldsymbol{\xi} \in U_{\alpha,Q}} \mathcal{I}\text{nfo}(\boldsymbol{\xi})$ d'après équation (4.27);
· $DOE = \{DOE, (\boldsymbol{\xi}^\star, \phi(\boldsymbol{\xi}^\star))\}$;
· Construire $\hat{\phi}$ à partir du *DOE* (avec son estimation d'erreur $\hat{\sigma}$);
· Calculer l'α-quantile $\tilde{q}_{\alpha_{\hat{\phi}-k\hat{\sigma}}}$ et le domaine de confiance $U_{\alpha,Q}$;
· Calculer la valeur du superquantile sur la borne inférieure et supérieure du krigeage $Q_{\hat{\phi}-k\hat{\sigma}}$ et $Q_{\hat{\phi}+k\hat{\sigma}}$;

end

La première fonction de test est la fonction de Rastrigin [Buttazzo et Frediani, 2009]. Elle est définie pour un nombre de dimension quelconque $d \in \mathbb{N}$ de la manière suivante :

$$\begin{cases} \phi(\boldsymbol{\xi}) &= 10d + \sum_{i=1}^{d} \left[\xi_i^2 - 10\cos(2\pi\xi_i)\right] \\ \boldsymbol{\xi} &\sim \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d) \end{cases} \tag{8.17}$$

Cette fonction est représentée pour $d = 2$ sur la figure 8.4. Nous pouvons remarquer

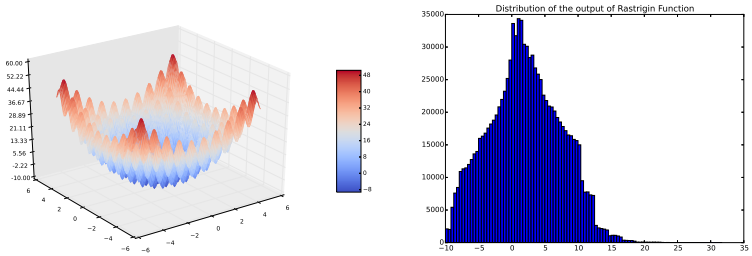


FIGURE 8.4 – Visualisation de la fonction Rastrigin et de la densité de probabilité de sa sortie pour $d = 2$

qu'elle est fortement multimodale, ce qui signifie que les zones de l'espace des entrées qui participent au calcul du superquantile sont disjointes. Cette caractéristique permet en particulier d'éprouver la méthode sur un cas sur lequel elle n'est *a priori* pas la plus adaptée.

Un second cas test est la fonction à quatre branches provenant de [Morio et Balesdent, 2015]. Elle est définie en dimension 2 et a pour équation :

$$\begin{cases} \phi\left(\xi_1, \xi_2\right) & = & 10 - \begin{cases} 3 + 0.1(\xi_1 - \xi_2)^2 - \frac{\xi_1 + \xi_2}{\sqrt{2}} \\ 3 + 0.1(\xi_1 - \xi_2)^2 + \frac{\xi_1 + \xi_2}{\sqrt{2}} \\ (\xi_1 - \xi_2) + \frac{7}{\sqrt{2}} \\ (\xi_2 - \xi_1) + \frac{7}{\sqrt{2}} \end{cases} \\ \boldsymbol{\xi} & \sim & \mathcal{N}(\mathbf{0}_2, \boldsymbol{I}_2) \end{cases} \tag{8.18}$$

Cette fonction est représentée sur la figure 8.5. Elle est constituée de quatre modes qui

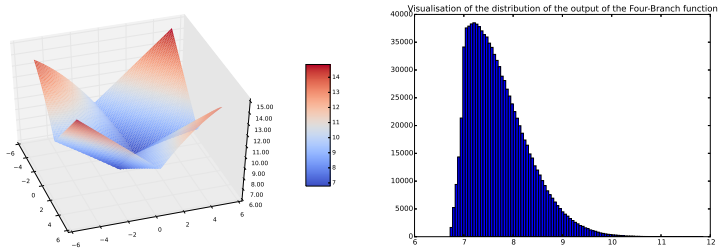


FIGURE 8.5 – Visualisation de la fonction à quatre branches et de la densité de probabilité de sa sortie

peuvent participer à l'estimation du superquantile. Elle présente l'avantage de ne pas être dérivable bien que continue et donc pas infiniment régulière.

Le troisième cas test provient également de [Morio et Balesdent, 2015] et s'appelle *Polynomial Product Function* (ici appelée PPF). Elle est également utilisable en dimension $d \in \mathbb{N}^\star$ quelconque et est définie de la manière suivante :

$$\begin{cases} \phi\left(\boldsymbol{\xi}\right) & = & \frac{1}{2}\sum_{i=1}^{d}(\xi_i^4 + \xi_i^2 + 5\xi_i) \\ \boldsymbol{\xi} & \sim & \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d) \end{cases} \tag{8.19}$$

Elle est représentée sur la figure 8.6 pour $d = 2$. Comme la fonction à quatre branches,

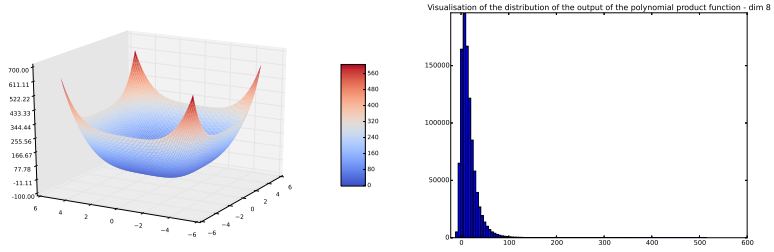


FIGURE 8.6 – Visualisation de la fonction PPF et de la densité de probabilité de sa sortie pour $d = 2$

elle est multimodale, les modes d'intérêt se trouvant au bord du domaine.

Enfin, la dernière fonction de test est la fonction du problème MOO analytique présenté

à l'équation (7.9) dont les entrées sont incertaines.

$$\begin{cases} \phi(\boldsymbol{\xi}) = \sqrt{|\xi_1 + \xi_2|}\left(1 - \exp\left[-\sum_{i=1}^{d}\left(\xi_i - \frac{1}{\sqrt{d}}\right)^2\right]\right) \\ \boldsymbol{\xi} \sim \mathcal{N}_{\mathcal{D}_\xi}(\boldsymbol{z}, \boldsymbol{\sigma}) \\ \boldsymbol{z} = (\boldsymbol{x}, 0.0, 0.0, 0.0, 0.0) \text{ et } \boldsymbol{x} \in \mathcal{D}_x \subset \mathbb{R}^4 \\ \boldsymbol{\sigma} = (2, 0.5, 0.3, 0.004, 0.05, 0.7, 0.05, 1.9) \end{cases} \tag{8.20}$$

Les lois normales utilisées pour ce cas test sont des gaussiennes tronquées, définies en annexe A.2.3. Le domaine de variation des incertitudes, permettant de tronquer la loi, est donné ci-après. Il s'agit en fait du même que pour le cas test pire cas du chapitre 7.

$$\text{pour } \boldsymbol{\xi}_x \ : \begin{cases} -2.5 \leqslant \xi_{x_1} \leqslant 2.5 \\ -0.8 \leqslant \xi_{x_2} \leqslant 0.8 \\ -0.9 \leqslant \xi_{x_3} \leqslant 0.9 \\ -0.01 \leqslant \xi_{x_4} \leqslant 0.01 \end{cases} \qquad \text{pour } \boldsymbol{\xi}_e \ : \begin{cases} -0.15 \leqslant \xi_{e_1} \leqslant 0.15 \\ -1.2 \leqslant \xi_{e_2} \leqslant 1.2 \\ -0.125 \leqslant \xi_{e_3} \leqslant 0.125 \\ -3.0 \leqslant \xi_{e_1} \leqslant 3.0 \end{cases}$$

La fonction est représentée sur la figure 7.1 pour $N_x = 2$ et $N_{\xi_e} = 0$. On remarque qu'elle est non-linéaire mais non multimodale. C'est d'ailleurs le cas qui ressemble le plus à notre cas industriel : une fonction régulière où les incertitudes sont imposées localement autour de divers points $\boldsymbol{x}$.

8.2.3.2 Test de l'algorithme sans modèle de substitution

Cette partie a pour but de valider l'algorithme 8.1 sur les cas tests analytiques introduits précédemment. Pour rappel, dans le cas industriel, nous nous intéressons à l'estimation du superquantile à 95%. On pose donc $\alpha = 0.95$. Afin d'appliquer cet algorithme, il est nécessaire de fixer trois paramètres : la taille de l'échantillon à chaque étape N_{CE}, le seuil β des quantiles empiriques intermédiaires et la loi biaisée utilisée. Les deux premiers sont liés : le choix de ces deux paramètres se fait de manière conjuguée puisqu'ils permettent de choisir le nombre de points qui est utilisé à chaque itération afin de calculer les nouveaux paramètres optimaux. En ce qui concerne le second, il permet de prendre en compte l'*a priori* que l'on peut avoir sur la fonction : dans le cas général, la loi baisée est prise dans la famille des lois normales. Cela est dû au fait que la résolution du problème de l'équation (8.9) est analytique dans cette famille de loi. La moyenne et la variance de la loi normale sont optimisées simultanément dans chaque cas.

En ce qui concerne la connaissance des 95%-superquantiles de référence sur ces cas tests, nous les calculons par une méthode de Monte-Carlo brut avec un échantillon de taille très élevée ($N_{MC} = 10^7$). Cela ne pose pas de problème ici puisque les fonctions étudiées sont connues analytiquement. Enfin, puisque l'algorithme d'entropie croisée n'est pas déterministe, les tableaux de résultat montrés ici sont calculés sur 100 répétitions.

Le tableau 8.1 récapitule les valeurs choisies pour l'application de l'algorithme 8.1. Ce tableau donne également un premier résultat de calcul : le nombre d'itérations maximal utilisé lors des 100 répétitions. On remarque que le nombre d'itérations maximum varie avec la complexité du cas test et le choix des valeurs N_{CE} et β. Ces deux paramètres permettent de contrôler le nombre d'échantillons utilisés pour calculer les paramètres optimaux à chaque itération de l'algorithme 4.2, c'est-à-dire n'annulant pas l'indicatrice de la formule (4.23). Ce nombre est égal à $(1 - \beta)N_{CE}$. Ainsi dans le cas général, plus cette

	N_{CE}	β	Nombre itérations max (résultat de calcul)
4 branches	100	0.60	10
Rastrigin	100	0.60	10
MOP2 modifié	100	0.70	8
PPF	400	0.70	4

TABLEAU 8.1 – Récapitulatif des paramètres choisis pour les cas tests utilisés.

quantité est réduite, plus le nombre d'itérations nécessaires doit être important. On remarque que pour les cas tests en dimension 2 (quatre branches et Rastrigin), on utilise 40 échantillons à chaque itération afin de déterminer les paramètres optimaux à l'algorithme 4.2 et 8.1. Ces cas tests sont multimodaux et il est donc nécessaire d'utiliser une taille d'échantillon plus importante que pour le cas test MOP2 modifié, où l'on se limite à 30. Enfin, pour le cas test PPF, il est multimodal et en dimension 8, nous avons donc choisi d'utiliser une taille d'échantillon égal à 120 pour le calcul des paramètres optimaux.

Les tableaux de résultat 8.2, 8.3, 8.4 et 8.5 permettent de comparer le résultat de l'algorithme 8.1 avec une méthode de Monte-Carlo utilisant un échantillon de même taille et le même algorithme mais sans la formule adaptée au superquantile. Chacun des tableaux correspond à un cas test différent. On remarque que quelle que soit la dimension, l'algorithme d'entropie croisée modifié permet de diminuer l'erreur commise sur l'estimation du superquantile à 95% par rapport aux deux autres. L'amélioration est nette par rapport à une méthode de Monte-Carlo crue utilisant le même nombre d'appels à ϕ. Pour insister sur ce point, nous avons également recherché la taille de l'échantillon Monte-Carlo permettant d'obtenir un résultat équivalent. Les deux boîtes à moustaches de gauche des figures 8.7, 8.8, 8.9 et 8.10 représentent la distribution de l'erreur relative des méthodes d'entropie croisée et de Monte-Carlo crue selon le nombre d'appels à ϕ. Le tableau 8.6 récapitule ces boîtes à moustache et met en évidence que l'algorithme d'entropie croisée introduit permet de diminuer de manière importante le nombre d'appels à ϕ pour estimer un superquantile à 95%. L'algorithme d'entropie croisée nécessite entre 2.5 à 4 fois moins d'appels à ϕ afin de fournir un résultat équivalent. En particulier, on remarque que plus la distribution de la sortie a une queue lourde, plus le gain par rapport à une méthode de Monte-Carlo classique est grand. En particulier, pour le cas test à quatre branches, la figure 8.5 illustre que la queue de distribution de la sortie est lourde. On remarque sur le tableau 8.6 que le gain de l'algorithme 8.1 est ici important. La même remarque peut être faite pour le cas test PPF (voir la figure 8.6). À l'inverse, pour le cas test Rastrigin, la densité de la sortie est donnée en figure 8.4. On voit que la queue de cette distribution n'est pas très *étalée*. Cette fois-ci le gain par rapport à des méthodes de Monte-Carlo est moins marqué, bien que toujours présent. Un autre problème que peut poser le cas test Rastrigin pour l'algorithme d'entropie croisée présenté ici est le grand nombre de maxima locaux de ϕ disjoints dans $\mathcal{D}_x$ qui participent au calcul du superquantile. En effet, la loi biaisée sur les entrées utilisée est une loi normale, il est donc difficile de favoriser les évènements de ces maxima locaux par le choix de la moyenne et de la variance dans chaque dimension d'entrée.

Le calcul des paramètres optimaux par la formule adaptée au superquantile de l'équation (8.8) permet d'améliorer légèrement le résultat par rapport à la formule adaptée au

	Algorithm 8.1 with equation (4.23)	Algorithm 8.1	Crude Monte-Carlo
Calls to ϕ	1000	1000	1000
Mean Error	3.4×10^{-3}	3.2×10^{-3}	6.4×10^{-3}
$q_{95\%}$ Error	7.7×10^{-3}	7.2×10^{-3}	1.5×10^{-2}
Max Error	1.0×10^{-2}	9.3×10^{-3}	2.6×10^{-2}

TABLEAU 8.2 – Valeur absolue de l'erreur relative commise dans l'estimation du 95%-superquantile pour le cas test à quatre branches. Ces valeurs sont calculées sur une répétition de 100 estimations pour chacune des méthodes.

	Algorithm 8.1 with equation (4.23)	Algorithm 8.1	Crude Monte-Carlo
Calls to ϕ	1000	1000	1000
Mean Error	1.8×10^{-2}	1.7×10^{-2}	2.5×10^{-2}
$q_{95\%}$ Error	4.7×10^{2}	4.3×10^{-2}	5.7×10^{-2}
Max Error	6.4×10^{-2}	5.8×10^{-2}	8.8×10^{-2}

TABLEAU 8.3 – Valeur absolue de l'erreur relative commise dans l'estimation du 95%-superquantile pour le cas test Rastrigin. Ces valeurs sont calculées sur une répétition de 100 estimations pour chacune des méthodes.

quantile de l'équation (4.23). Néanmoins, cette amélioration n'est pas significative. Cela est dû au fait que l'effectif N_{CE} est faible dans chacun de ces cas. Le calcul des paramètres $\boldsymbol{\lambda}$ optimaux de la dernière itération de l'algorithme 8.1 utilise donc très peu de réalisations de $\boldsymbol{\xi}$, ce qui explique le peu de différence dans les résultats.

On compare ensuite les résultats en augmentant N_{CE} à 1000 par itération dans un cas où la queue de distribution de la sortie de la fonction est lourde. Par exemple pour le cas PPF (la figure 8.6 montre la densité de la sortie), l'adaptation de la formule pour la maximisation du $\boldsymbol{\lambda}$ apporte une amélioration visible au niveau de la boîte à moustache de l'erreur relative, comme l'illustre la figure 8.11. Le résultat de cette figure a été lancé avec le même paramètre qu'au tableau 8.1, l'unique modification est la taille N_{CE} d'échantillons par itération qui passe de 400 à 1000.

8.2.3.3 Test de l'algorithme sur la prédiction par krigeage

Dans cette section, nous montrons que l'utilisation du krigeage (détaillée à la section 8.2.2) permet de diminuer de manière importante le nombre d'appels nécessaires à la fonction ϕ. Puisque nous n'utilisons plus la fonction de référence, nous devons aussi illustrer que l'erreur commise dans l'estimation du superquantile n'est pas très impactée par l'utilisation d'un modèle de substitution.

Pour cela, nous reprenons les paramètres détaillés au tableau 8.1 et nous appliquons l'algorithme 8.2 à chaque étape de l'algorithme 8.1 où la valeur de ϕ doit être connue au-delà du superquantile. Ceci signifie que cet enrichissement est réalisé avant la résolution de l'équation (8.8) et avant l'estimation finale du superquantile.

Pour fixer le critère d'arrêt de l'enrichissement détaillé à l'algorithme 8.2, nous avons choisi d'utiliser l'écart relatif entre le superquantile estimé sur la borne supérieure du krigeage et celui estimé sur la borne inférieure. L'algorithme s'arrête si cette différence

	Algorithm 8.1 with equation (4.23)	Algorithm 8.1	Crude Monte-Carlo
Calls to ϕ	1600	1600	1600
Mean Error	2.2×10^{-2}	1.9×10^{-2}	4.4×10^{-2}
$q_{95\%}$ Error	5.0×10^{-2}	4.4×10^{-2}	10.8×10^{-2}
Max Error	7.7×10^{-2}	6.6×10^{-2}	14.6×10^{-2}

TABLEAU 8.4 – Valeur absolue de l'erreur relative commise dans l'estimation du 95%-superquantile pour le cas test PPF en dimension 8. Ces valeurs sont calculées sur une répétition de 100 estimations pour chacune des méthodes.

	Algorithm 8.1 with equation (4.23)	Algorithm 8.1	Crude Monte-Carlo
Calls to ϕ	700	700	700
Mean Error	6.0×10^{-3}	5.9×10^{-3}	8.1×10^{-3}
$q_{95\%}$ Error	1.9×10^{-2}	1.9×10^{-2}	2.4×10^{-2}
Max Error	9.0×10^{-2}	4.0×10^{-2}	9.2×10^{-2}

TABLEAU 8.5 – Valeur absolue de l'erreur relative commise dans l'estimation du 95%-superquantile pour le cas test MOP2 modifié en dimension 8. Ces valeurs sont calculées sur une répétition de 10 estimations en 100 points LHS tirés dans l'espace $\mathcal{D}_x$.

Cas test	quatre branches	Rastrigin	PPF	MOP2 modifié
N_{MC}	4000	2500	6000	1800
N_{CE} total	1000	1000	1600	700

TABLEAU 8.6 – Tableau récapitulatif de la taille du Monte-Carlo à utiliser afin d'obtenir le même résultat que par l'algorithme 8.1 d'entropie croisée.

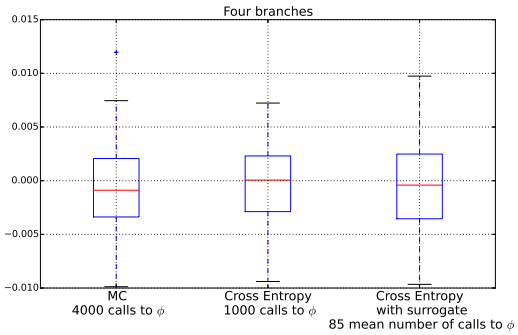


FIGURE 8.7 – Boîtes à moustaches de l'erreur relative sur l'estimation du superquantile.

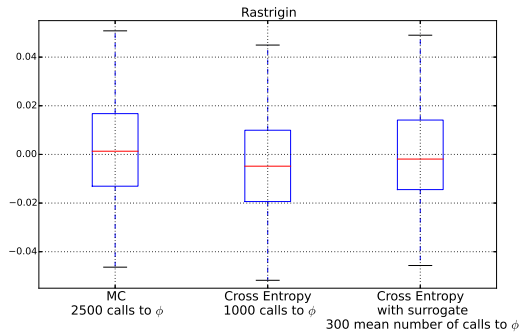


FIGURE 8.8 – Boîtes à moustaches de l'erreur relative sur l'estimation du superquantile.

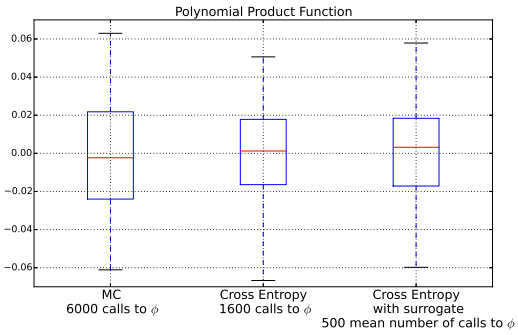


FIGURE 8.9 – Boîtes à moustaches de l'erreur relative sur l'estimation du superquantile.

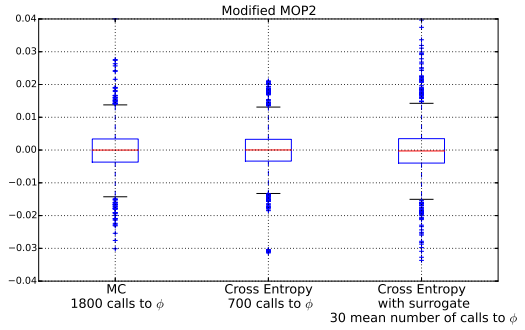


FIGURE 8.10 – Boîtes à moustaches de l'erreur relative sur l'estimation du superquantile.

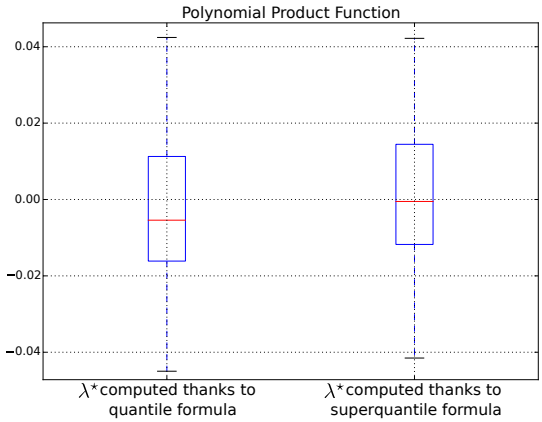


FIGURE 8.11 – Boîtes à moustaches de l'erreur relative sur l'estimation du superquantile. Comparaison entre l'utilisation de l'équation (4.23) et l'équation (8.8) pour le calcul de $\boldsymbol{\lambda}^\star$ à l'algorithme 8.1 avec $N_{CE} = 1000$ à chaque itération.

relative est plus petite que 10^{-2}. Les résultats obtenus sont détaillés sur les deux boîtes à moustache de droite des figures 8.7, 8.8, 8.9 et 8.10. On y remarque d'abord que les boîtes à moustache sont très similaires que ce soit avec ou sans modèles de substitution. Ceci met en évidence que l'enrichissement du plan d'expériences du modèle de substitution est efficace.

Ensuite, en légende de ces graphiques, le nombre (moyen) d'appels nécessaires à la fonction ϕ est donné, avec et sans modèle de substitution. Ces résultats sont résumés dans le tableau 8.7.

Cas test	Quatre branches	Rastrigin	PPF	MOP2 modifié
Nombre d'appels à ϕ sans métamodèle	1000	1000	1600	700
Nombre moyen d'appels à ϕ **avec** métamodèle	85	300	500	30

TABLEAU 8.7 – Tableau comparant le nombre d'appels nécessaires à ϕ entre l'algorithme 8.1 assisté par modèle de substitution (comme à l'algorithme 8.2) ou non.

On remarque que l'efficacité de l'utilisation d'un modèle de substitution est variable selon les cas. Par exemple, on remarque que pour la fonction PPF en dimension 8 ainsi que pour Rastrigin, le modèle de substitution permet de ne calculer qu'un tiers des points, ce qui reste important. Ceci s'explique par le fait qu'il y a plusieurs zones de $\mathcal{D}_x$ à connaître avec précision. Or, pour l'estimation du superquantile, plus il y a de zones de l'espace des entrées qui participent à son calcul (ce qui est le cas de Rastrigin et de PPF comme les figures 8.4 et 8.6 le montrent), plus il y a de zones à connaître avec précision. Il est donc nécessaire d'enrichir davantage le DOE du modèle de krigeage.

De plus, ces fonctions ne sont pas "stationnaires". C'est l'une des hypothèses de la construction par krigeage introduite à la section 3.2.3. Elle suppose qu'il existe une fonction r telle que $\mathrm{Cov}(Z(\boldsymbol{x}), Z(\boldsymbol{x}+\boldsymbol{h})) = r(\boldsymbol{h})$. En d'autres termes, la matrice de covariance ne dépend pas de la zone de l'espace des entrées où l'on souhaite prédire la fonction ϕ. Cela suppose que la fonction ϕ a des variations qui sont comparables dans chaque domaine de l'espace $\mathcal{D}_x$. Or, cette condition est difficilement vérifiée sur PPF et Rastrigin.

8.2.4 Conclusions sur l'algorithme d'entropie croisée pour l'estimation de superquantile

Pour résumer, l'algorithme 8.1 permet :

— de diminuer le nombre d'appels nécessaires à ϕ pour estimer le superquantile avec suffisamment de précision par rapport à une méthode de Monte-Carlo crue. Le gain varie selon les cas tests. Le tableau 8.6 compare le nombre d'appels à ϕ pour obtenir la même erreur sur l'estimation du superquantile. Il est notamment plus intéressant d'utiliser cet algorithme dans le cas où la queue de distribution de la sortie est lourde, ce qui est le cas de certaines zones de l'espace des entrées $\boldsymbol{x} \in \mathcal{D}_x$ du problème industriel.

— de diminuer l'erreur d'estimation sur le superquantile par rapport à l'algorithme 8.1 n'utilisant pas la formule modifiée pour le superquantile (de l'équation (8.8)) introduite à la section 8.2.1.1. Toutefois, les tableaux de résultats 8.2, 8.3, 8.4 et 8.5

montrent que cette amélioration n'est pas très importante, même en augmentant le nombre d'échantillons N_{CE} utilisés par la méthode.

De plus, l'assistance par krigeage proposée à l'algorithme 8.2 permet de diminuer le nombre d'appels nécessaires à ϕ, tout en ne rajoutant pas, voire très peu, d'erreur à l'estimation du superquantile. Le tableau 8.7 récapitule l'apport du krigeage dans le nombre d'appels nécessaires à ϕ afin de converger vers une estimation du superquantile suffisante.

En ce qui concerne les limites de l'algorithme, on peut remarquer que :

— les paramètres N_{CE} et β peuvent être compliqués à choisir *a priori*. Pour rappel, ils contrôlent le nombre d'échantillons utilisés pour calculer les paramètres optimaux à chaque itération de l'algorithme 4.2, c'est-à-dire n'annulant pas l'indicatrice de la formule (4.23). Ce nombre est égal à $(1-\beta)N_{CE}$. Toutefois, nous pouvons vérifier une fois l'algorithme terminé que l'erreur n'est pas trop grande. À l'aide de l'estimateur de la variance asymptotique de l'erreur commise par l'estimation par échantillonnage préférentiel sur le superquantile fourni par le théorème 4.2, nous avons un estimateur d'erreur théorique qui peut servir pour déterminer un critère de décision. Si cette variance approchée calculée à la fin de l'algorithme 8.1 n'est pas satisfaisante, nous pouvons augmenter la taille N_{CE} et relancer l'algorithme. De plus, il est possible de réutiliser l'échantillon généré lors de l'exécution non satisfaisante en les ajoutant à l'échantillon initial de l'algorithme 8.1, ou en reprenant le plan d'expériences final du modèle de krigeage utilisé.

— si la fonction ϕ a trop de zones disjointes dans l'espace des entrées qui participent à l'estimation du superquantile (donc si elle est multimodale), la méthode peut devenir coûteuse, voire inadaptée. L'une des raisons à cela est que la recherche d'une "bonne" loi biaisée peut devenir problématique. En particulier, la loi normale utilisée ici n'est plus forcément adaptée.

— si la dimension d'entrée est trop grande, les limitations inhérentes au krigeage (voir la fin de la section 6.4.5) limiteront les performances de l'algorithme.

8.3 Mise en place de l'optimisation multi-objectif probabiliste

Dans cette partie, nous appliquons l'algorithme MOEGO NSGA-II au problème (8.1). Pour cela, nous avons prouvé au théorème 8.1 que la fonction $g_\rho(\boldsymbol{x}) = \rho_{\boldsymbol{\xi}}\left[f\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)\right]$ est continue sous certaines conditions sur la mesure de risque ρ_ξ. En particulier, le superquantile vérifie ces conditions.

De plus, nous avons développé un algorithme d'estimation de superquantile basé sur l'entropie croisée et le krigeage qui limite le nombre d'appels à la fonction ϕ et conduit à une estimation précise du superquantile. Il est particulièrement intéressant ici puisque très adapté à l'estimation de superquantile dans le cas de fonctions ne possédant pas un grand nombre de maxima locaux. Or, le problème industriel entre dans ce cas puisque la fonction de durée de vie est très régulière et les incertitudes entraînent des variations très localisées dans l'espace des entrées.

Enfin, comme pour le pire cas à la section 7.1.3, il peut être intéressant de réutiliser les appels à la fonction f pour estimer le superquantile en divers points de contrôle $\boldsymbol{x}$ au cours de l'optimisation par MOEGO NSGA-II. Deux procédures sont alors possibles :

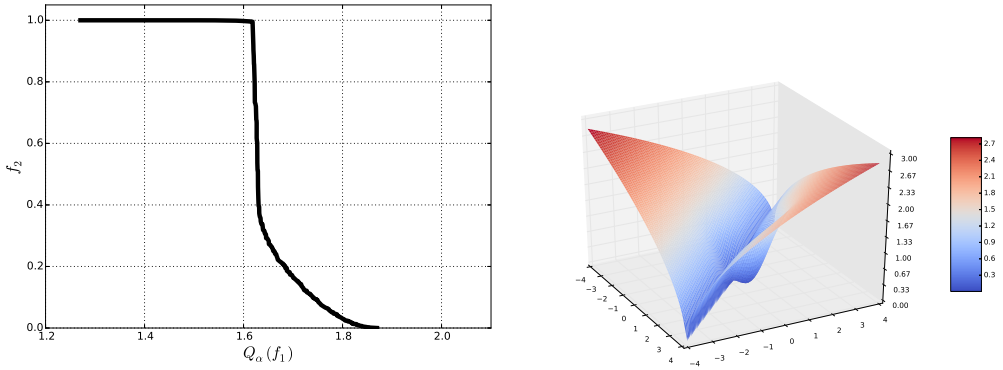


FIGURE 8.12 – Illustration du $PF^\star$ dans le cas où $N_x = 4$ et $N_\xi = 8$ et de l'objectif incertain dans le cas où $N_x = 2$ et $N_{\xi_e} = 0$

1. La première est la même que celle de la section 7.1.3 : on peut rajouter, par l'algorithme 7.1, les points de Ξ_x (défini à l'équation (7.8)) au plan d'expériences initial de l'estimation du superquantile. En d'autres termes, ceci signifie que nous pouvons rajouter les appels déjà réalisés à la fonction f dans le plan d'expériences initial de l'algorithme 8.2 permettant d'estimer le superquantile autour d'un point $\boldsymbol{x}$ quelconque (en posant $\phi(\boldsymbol{\xi}) = f(\boldsymbol{\zeta}(\boldsymbol{x}, \boldsymbol{\xi}_x), \boldsymbol{\xi}_e)$).
2. La seconde est d'utiliser les points de Ξ_x dans l'échantillon aléatoire initial des algorithmes 4.2 et 8.1. Toutefois, ceci nécessite d'être capable d'associer une densité de probabilité avec laquelle les entrées incertaines de Ξ_x ont été générées, ce qui n'est pas toujours possible. En effet, si ces points ne sont pas générés selon une loi de probabilité connue, il peut être très coûteux (et donc inenvisageable) de l'estimer. C'est en particulier le cas ici puisque les points de Ξ_x proviennent de l'exécution de l'algorithme 8.2 pour l'enrichissement du modèle de krigeage. Ces points ne coïncident pas forcément avec l'échantillon Monte-Carlo généré dans l'algorithme 8.1. Leur distribution n'est donc pas connue. Même si cette manière de réutiliser les appels à la fonction f n'est pas applicable ici, nous avons prouvé la convergence d'un estimateur par échantillonnage préférentiel utilisant des échantillons générés selon des lois différentes. Le lecteur intéressé peut se référer à la section 10.4 du chapitre 10.

Nous pouvons donc en conclure que la réutilisation proposée à l'algorithme 7.1 est la seule viable dans notre cadre. Nous l'appliquons en section 8.3.2 au cas test analytique introduit à la section 8.3.1. Il est ensuite testé en section 8.4 sur le cas industriel introduit en section 2.4.

8.3.1 Présentation du cas test analytique

Afin d'éprouver l'algorithme détaillé précédemment sur un cas test d'une complexité suffisante, nous reprenons le cas test MOP2 modifié détaillé à la section 7.1.2.4. L'équation des objectifs est la même qu'à l'équation 7.9. Toutefois, nous prenons en compte cette fois-ci la loi de probabilité des incertitudes afin de calculer l'objectif f_1. Ces incertitudes sont détaillées à l'équation (8.20).

Ce cas est utilisé dans la dimension du problème industriel, ce qui signifie $N_x = 4$ et

$N_\xi = 8$. Le domaine de variation est le suivant : $\forall i \in \{1, \cdots, N_x\}$, $-4 \leqslant x_i \leqslant 4$. En ce qui concerne les incertitudes $\boldsymbol{\xi}_x$ portant sur les paramètres de contrôle, elles sont de type additive. Ceci revient à définir la fonction $\boldsymbol{\zeta}$ de la manière suivante : $\zeta_i(x_i, \xi_i) = x_i + \xi_i$ pour toutes les composantes i de N_{ξ_x}. Enfin, pour le domaine de variation des incertitudes, nous avons :

$$\text{pour } \boldsymbol{\xi}_x \ : \begin{cases} -2.5 \leqslant \xi_{x_1} \leqslant 2.5 \\ -0.8 \leqslant \xi_{x_2} \leqslant 0.8 \\ -0.9 \leqslant \xi_{x_3} \leqslant 0.9 \\ -0.01 \leqslant \xi_{x_4} \leqslant 0.01 \end{cases} \qquad \text{pour } \boldsymbol{\xi}_e \ : \begin{cases} -0.15 \leqslant \xi_{e_1} \leqslant 0.15 \\ -1.2 \leqslant \xi_{e_2} \leqslant 1.2 \\ -0.125 \leqslant \xi_{e_3} \leqslant 0.125 \\ -3.0 \leqslant \xi_{e_1} \leqslant 3.0 \end{cases}$$

Ici ces bornes nous servent à définir la troncature des lois normales définies à l'équation 8.20. Les lois normales tronquées sont définies à la section A.2.3 de l'annexe A.

Remarque concernant l'estimation du $PF^\star$ de la figure 8.12 : afin d'avoir une référence à laquelle se comparer, nous avons appliqué sur le cas test MOP2 modifié l'algorithme NSGA-II (détaillé en section 5.2.1). Pour estimer le superquantile en chaque $\boldsymbol{x}$, nous avons utilisé la méthode de Monte-Carlo avec des échantillons de taille élevée ($N_{MC} = 10^6$). En répétant 50 fois cette optimisation et en gardant les points Pareto-optimaux de ces 50 répétitions, nous avons une approximation suffisamment précise de $PF^\star$.

8.3.2 Application au cas test analytique

8.3.2.1 Application avec une estimation du superquantile bruitée

Afin de valider notre approche, nous comparons ici le résultat de MOEGO NSGA-II couplé à l'entropie croisée de l'algorithme 8.1 à l'algorithme NSGA-II classique pour la partie multi-objectif de l'équation (8.1) et un Monte-Carlo de taille 10^5 pour estimer le 95%-superquantile en chaque point de contrôle multi-objectif $\boldsymbol{x}$. Cette résolution est volontairement lourde et donc coûteuse en nombre d'appels aux deux fonctions objectif. Le résultat de cette optimisation constitue en fait la référence à laquelle se comparer.

En ce qui concerne la paramétrisation de l'algorithme MOEGO NSGA-II, nous choisissons les paramètres détaillés à la figure 6.37. Pour le calcul du superquantile en chaque $\boldsymbol{x}$, les paramètres de l'algorithme 8.1 sont ceux du tableau 8.1. Comme les résultats de la partie 8.2.3 le montrent, notamment le tableau 8.5, l'estimation de la fonction $g(\boldsymbol{x}) = Q_{95\%}\left[f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)\right]$ est bruitée, avec un bruit moyen se situant entre 10^{-2} et 10^{-3}. Nous pouvons donc appliquer l'algorithme MOEGO NSGA-II à la manière de celle détaillée en section 6.4.4.1. C'est-à-dire que nous retournons le résultat de la population finale de l'algorithme MOEGO NSGA-II et non pas le front de Pareto du DOE final, correspondant à des appels provenant d'une fonction bruitée. Pour illustrer l'importance de l'utilisation de la dernière population plutôt que celle du DOE, les figures 8.13 et 8.14 montrent le front obtenu selon les deux méthodes La figure 8.13 met en évidence que le bruit provenant de l'erreur du superquantile affecte le front calculé sur le DOE, ce qui n'était pas le cas pour la méthode basée sur le pire cas que l'on approchait précisément. Le front n'est pas bien capté, notamment la partie où $Q_{95\%}(f_1) \simeq 1.81$. Maintenant, si l'on prend la dernière population convergée par NSGA-II sur les prédictions par krigeage à cette même itération et que l'on évalue ces points sur un Monte-Carlo (figure 8.14), le front obtenu est tout à fait satisfaisant. Afin de compenser les effets du bruit, un effet

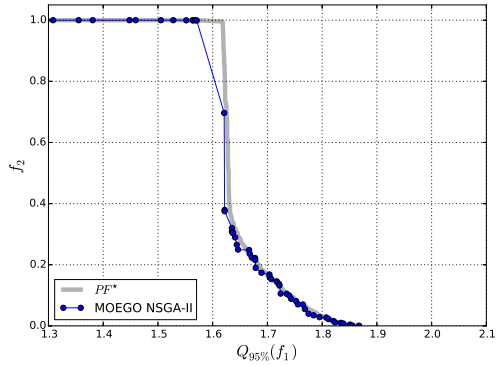


FIGURE 8.13 – Graphique représentant le front obtenu sur le DOE de MOEGO NSGA-II après 30 appels distribués.

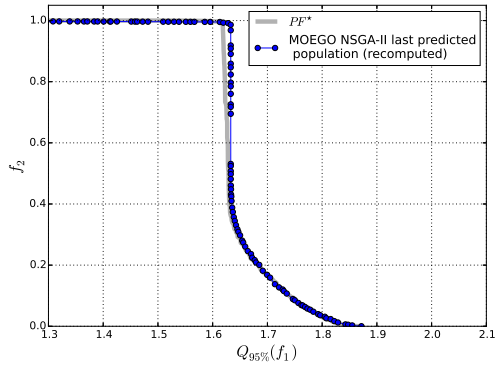


FIGURE 8.14 – Graphique représentant le front obtenu sur la dernière population convergée de MOEGO NSGA-II (recalculée sur un Monte-Carlo de taille 10^5)) après 30 appels distribués. La répétition et le nombre d'appels distribués sont les mêmes qu'à la figure 8.13

pépite dont la magnitude est de l'ordre du bruit (10^{-3}) est utilisé. Ceci montre que la prise en compte de fonctions objectif bruitées par MOEGO NSGA-II peut être effective. Toutefois, le dernier front devant être réestimé, ceci demande d'allouer un budget de calcul légèrement supérieur à l'utilisation plus classique de cet algorithme multi-objectif (celle détaillée à la section 6.4).

8.3.2.2 Application avec une estimation plus précise du superquantile

Pour se ramener à une utilisation plus classique de MOEGO NSGA-II, nous augmentant la taille d'échantillon utilisé dans l'algorithme d'entropie croisée 8.1 afin d'en diminuer le bruit d'estimation. Nous passons de $N_{CE} = 100$ à $N_{CE} = 500$ et nous augmentons donc la valeur du seuil des quantiles intermédiaires à $\beta = 0.85$. Ceci équivaut à l'utilisation de 75 points de l'échantillon pour chaque estimation des paramètres biaisés. Le nombre d'itérations nécessaires à la convergence dans l'algorithme 8.1 descend à 4, ce qui signifie une taille d'échantillon maximale de 2000. Toutefois, nous utilisons une approximation par krigeage, dont l'enrichissement est détaillé à l'algorithme 8.2.

Comme précédemment, nous répétons l'optimisation cinquante fois et nous comparons

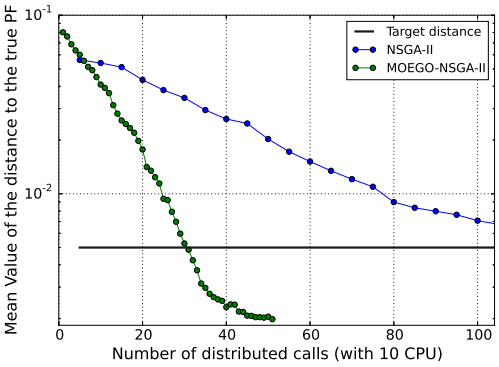


FIGURE 8.15 – Graphique représentant la moyenne sur cinquante optimisations de la distance entre $PF^\star$ et PF en fonction du nombre d'appels distribués aux fonctions objectif pour le cas test MOP2 modifié

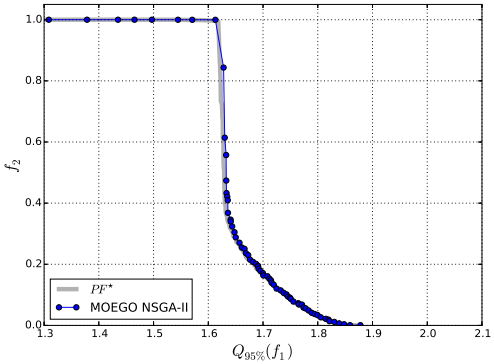


FIGURE 8.16 – Graphique représentant le front obtenu par MOEGO NSGA-II après 30 appels distribués et vérifiant le critère de l'équation (6.8) avec $\mathcal{M}^{target} = 0.005$.

la décroissance de la distance entre $PF^\star$ discrétisé et le front retourné par la méthode de référence (NSGA-II et Monte-Carlo) et la méthode proposée (MOEGO NSGA-II et entropie croisée). La figure 8.15 résume ce résultat. La distance cible choisie ici correspond à $\mathcal{M}^{target} = 0.005$, où la distance est celle détaillée à l'équation (6.6). La figure 8.16 illustre un front obtenu lorsque cette distance cible est atteinte.

On remarque que notre méthodologie nécessite un nombre restreint d'appels aux fonctions objectif afin d'atteindre une précision suffisante sur le front retourné. En particulier, la figure 8.17 met en évidence l'itération à partir de laquelle toutes les répétitions de l'algorithme ont atteint la distance cible. On remarque que MOEGO NSGA-II permet de converger en environ 38 appels distribués, alors que NSGA-II couplé à un Monte-Carlo en nécessite 155.

Nous souhaitons maintenant mettre en évidence l'apport de la réutilisation des appels à f_1 détaillée à l'algorithme 7.1. Pour cela, la figure 8.18 donne le nombre moyen d'appels à la fonction f_1 à chaque calcul de superquantile et pour chaque itération de l'algorithme MOEGO NSGA-II. Cette figure montre que la réutilisation a un effet positif sur le nombre d'appels nécessaires à la fonction f_1, puisque plus le nombre d'itérations est important, donc plus la base de données Ξ est remplie, moins il est nécessaire d'exécuter de simulations de f_1. Par ailleurs, ce graphique illustre, comme au chapitre 7 à la figure 7.5, que MOEGO

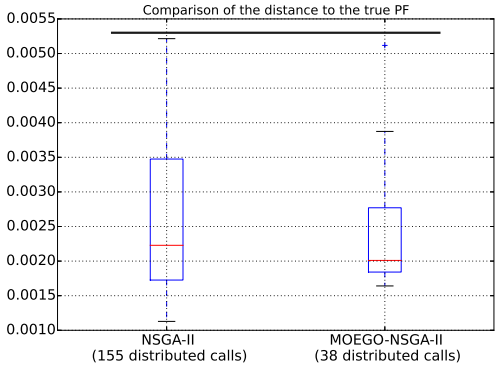


FIGURE 8.17 – Boîtes à moustaches du nombre d'appels aux fonctions objectif une fois le critère d'arrêt atteint pour le cas test MOP2 modifié

NSGA-II a une première phase à nombre d'appels réduit, une seconde phase où le nombre d'appels augmente beaucoup puis une phase finale qui nécessite moins d'appels. La zone où le nombre d'appels augmente traduit le fait que MOEGO NSGA-II explore intensivement l'espace $\mathcal{D}_x$. Ceci implique que les points $\boldsymbol{x}$ où l'on doit évaluer la fonction sont distants et donc que le cardinal de Ξ_x est nul. En d'autres termes, qu'il n'y a aucun point de la base de données Ξ qui se trouve à proximité des points de contrôle où l'on évalue le superquantile à ces itérations. Ceci met également en évidence que lors des premières itérations de l'algorithme MOEGO NSGA-II, l'exploration ne se met pas tout de suite en place et la méthode stagne dans une zone, ce qui justifie l'efficacité de la réutilisation.

Enfin, la fonction f_1 peut être coûteuse alors que nous lui appliquons le calcul d'un superquantile à chaque appel distribué. Pour quantifier le coût de résolution de l'équation (8.1), il faut regarder le nombre d'appels nécessaires à cette fonction à chaque itération de l'algorithme MOEGO NSGA-II. La figure 8.19 donne le nombre maximum d'appels à chaque itération de MOEGO NSGA-II. De cette dernière, on peut en déduire le temps d'exécution d'un tel algorithme. En effet, on remarque sur ce graphique qu'il faut en moyenne un maximum de 60 exécutions de f_1 afin de calculer le superquantile. Cela est dû en particulier à l'algorithme 7.1 de réutilisation. De ce fait, en conjuguant les 39 itérations de MOEGO NSGA-II et les 60 exécutions de f_1 à chaque itération, le temps d'exécution de l'algorithme est de l'ordre de 2300 exécutions de la fonction f_1 pour atteindre la convergence. Ce calcul se base sur la pire des cinquante répétitions réalisées. Ce qui nous donne donc une borne supérieure du nombre d'appels nécessaires.

8.3.2.3 Preuve de l'intérêt de l'optimisation du superquantile par rapport au pire cas

Un autre aspect qu'il est intéressant d'étudier ici est l'apport de la prise en compte des incertitudes sur les entrées d'un problème d'optimisation.

Pour cela, nous comparons les solutions des trois formulations introduites jusqu'à maintenant. La première est celle ne prenant pas en compte les incertitudes sur les différents paramètres du problème d'optimisation, c'est-à-dire celle donnée à l'équation (6.1). La deuxième est celle de l'équation (7.1) où les incertitudes sont prises en compte par un pire cas en chaque point de contrôle. La troisième est celle de l'équation (8.1) où les incertitudes sont prises en compte de manière probabiliste par le biais d'un 95%-superquantile.

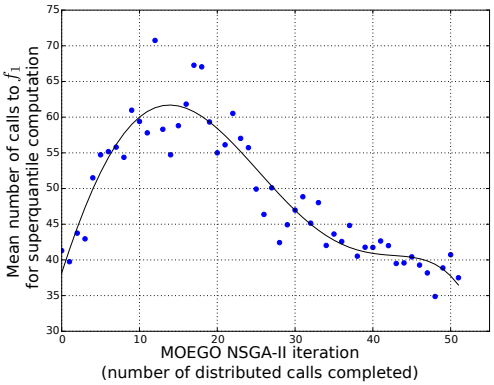


FIGURE 8.18 – Nombre moyen d'appels à f_1 à chaque calcul de superquantile pour chaque itération de l'algorithme MOEGO NSGA-II

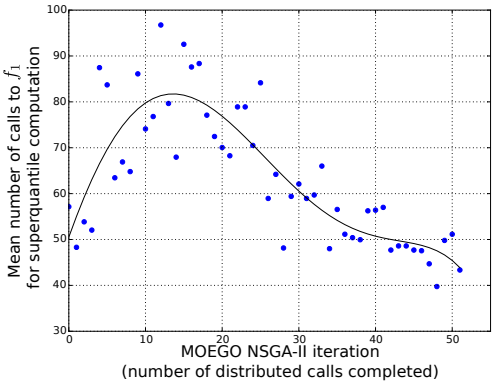


FIGURE 8.19 – Nombre maximum d'appels à f_1 à chaque calcul de superquantile pour chaque itération de l'algorithme MOEGO NSGA-II

Nous effectuons cette comparaison sur le cas test introduit à l'équation (8.20). La figure 8.20 donne sur un même graphique les fronts solutions de ces trois formulations. La forme du front superquantile est comparable à celle du front pire cas. On remarque également que la valeur de l'objectif f_1 est inférieure pour le front superquantile, ce qui est évident étant donné que le superquantile est une relaxation du pire cas. Pour comparer les solutions pire cas et superquantile, nous calculons ensuite le 95%-superquantile sur l'objectif f_1 des points du PF pire cas. Le résultat est donné à la figure 8.21. On remarque que les points solutions pour le superquantile ne sont pas très différents des points optimaux pour le pire cas. L'unique différence pourrait concerner les points situés sur la ligne d'équation $f_1 \simeq 1.81$. Mais ces points ne sont pas d'un très grand intérêt : ils augmentent beaucoup le critère f_2 sans diminuer de manière importante le critère f_1.

8.4 Application au cas industriel

Nous pouvons à présent appliquer l'algorithme MOEGO NSGA-II couplé à l'algorithme d'entropie croisée présenté dans la section 8.2.1.3 dans le cadre du cas test industriel. Le lien entre les deux est réalisé par l'algorithme 7.1. Le cas industriel étant proche de celui du cas analytique, par construction, les différents algorithmes utilisés (MOEGO NSGA-II et

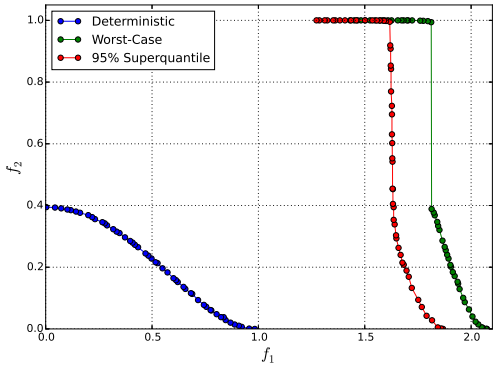


FIGURE 8.20 – Représentation dans l'espace des objectifs des points Pareto-optimaux pour la formulation déterministe, pour la formulation avec un pire cas et un superquantile pour quantifier les effets des incertitudes en entrée.

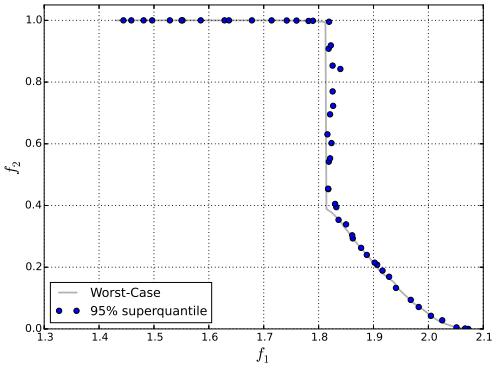


FIGURE 8.21 – Représentation du $Q_{95\%}\left[f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x}, \boldsymbol{\xi}_x\right), \boldsymbol{\xi}_e\right)\right]$ en fonction de f_2 des points Pareto-optimaux pour la formulation avec un pire cas et avec un superquantile pour quantifier les effets des incertitudes en entrée.

l'entropie croisée) et leur couplage (algorithme 7.1) sont paramétrés de la même manière.

8.4.1 Résultat de l'optimisation multi-objectif d'un superquantile

La figure 8.22 compare les fronts obtenus par MOEGO NSGA-II après 30 itérations et 50 itérations de cet algorithme. On remarque une fois encore que la différence est faible et que la convergence obtenue après 30 itérations est suffisante. Dès lors, nous pouvons en déduire le nombre d'appels nécessaires à la fonction coûteuse f_1 afin de retourner un front de Pareto de précision suffisante. Pour cela, la figure 8.23 représente le nombre d'appels maximal à la fonction f_1 pour le calcul du superquantile par l'algorithme 8.1 à chaque itération de MOEGO NSGA-II. Comme la tendance en noir le met en évidence, le nombre d'appels nécessaires décroît avec le nombre d'itérations, signe que la réutilisation est effective. Par ailleurs, on remarque que le nombre maximal d'appels à la fonction f_1 est le plus grand lors de ces premières itérations. Ceci illustre que les points de contrôle $\boldsymbol{x}$ où sont évalués les superquantiles ne sont pas proches dans l'espace des entrées. Cette figure met donc également en évidence le fait que l'exploration est effective dès les premières itérations de NSGA-II.

8.4.2 Comparaison avec le front pire cas et le front déterministe

Comme pour le cas test analytique, nous quantifions dans cette partie l'intérêt d'avoir pris en compte les incertitudes au niveau des paramètres de contrôle $\boldsymbol{x}$ des solutions obtenues. Pour cela et comme au chapitre 7, nous comparons les solutions des trois formulations (déterministe, pire cas, superquantile). La première est celle ne prenant pas en compte les incertitudes sur les différents paramètres du problème d'optimisation, c'est-à-dire celle donnée à l'équation (6.1). La deuxième est celle de l'équation (7.1) où les incertitudes sont prises en compte par un pire cas en chaque point de contrôle. La troisième est la solution de l'équation (8.1) où les incertitudes sont traitées de manière probabiliste à l'aide d'un 95%-superquantile.

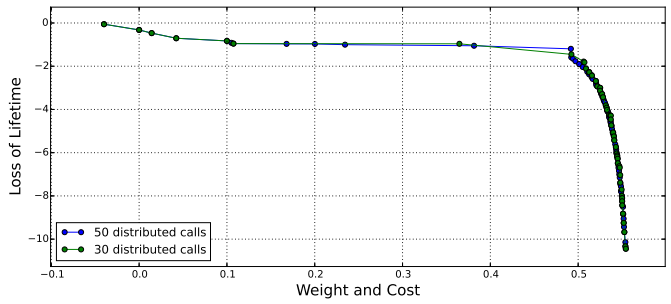


FIGURE 8.22 – Comparaison du front de Pareto obtenu par MOEGO NSGA-II couplé avec l'algorithme d'entropie croisée assisté par krigeage après 30 itérations de MOEGO NSGA-II et après 50 itérations de MOEGO NSGA-II.

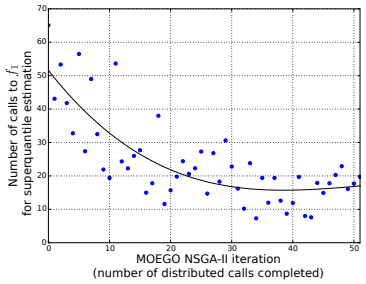


FIGURE 8.23 – Nombre maximal d'appels à f_1 réalisés par l'algorithme d'entropie croisée assisté par krigeage à chaque itération de l'algorithme MOEGO NSGA-II.

La figure 8.24 illustre d'abord l'écart qu'il y a entre ces trois fronts de Pareto. On remarque que la distance entre les solutions Pareto-optimales pour le pire cas, pour le superquantile et pour la formulation déterministe n'est pas constante quelle que soit la valeur de f_2. En effet, sur la partie gauche du front où $f_2 < 0.3$, on remarque que l'écart entre ces trois ensembles de point est pratiquement constant. Toutefois, si l'on regarde dans la zone où $f_2 > 0.45$, l'écart n'est plus aussi facilement quantifiable. Comme nous l'avions remarqué à la section 7.2.2, ceci est le signe que la densité de la sortie f_1 pour les solutions Pareto-optimales a un support plus ou moins grand et une queue plus ou moins lourde selon sa position dans l'espace des objectifs.

La figure 8.25 met en évidence que la résolution de l'équation 8.1 permet de trouver des solutions que nous n'aurions pas pu obtenir par la formulation avec un pire cas ou la formulation sans incertitudes. En détaillant les points *déterministes* (en vert sur la figure), on remarque que la résolution ne prenant pas en compte les incertitudes ne permet pas de trouver tout un ensemble de points qui correspondent pourtant à des solutions innovantes, puisque minimisant la perte de durée de vie. C'est la même conclusion qu'au chapitre précédent à la section 7.2.2.

En se concentrant sur les points optimaux pour la résolution pire cas, nous voyons qu'ils n'appartiennent pas tous au front de Pareto optimal lorsque l'on calcule leur 95%-superquantile. Nous avons également calculé la densité de la sortie en ces deux points extrêmaux pour chacun de ces deux fronts. Le résultat est donné à la figure 8.26. Nous remarquons que la solution optimale pour le superquantile a une distribution dont le support est plus grand que le point optimal pour le pire cas. L'une des justifications de l'utilisation d'un superquantile était en particulier de pouvoir proposer des solutions dont le pire cas peut être certes élevé mais peu probable. C'est ce que nous remarquons ici.

8.4.3 Interprétation physique du résultat

Puisque la forme du front est la même que pour la résolution pire cas et déterministe, le classement des solutions retournées réalisé à la section 7.2.3 reste valable dans cette partie. Toutefois, on peut ici rajouter une remarque grâce au résultat des figures 8.26 et 8.25. Ces dernières permettent de justifier de l'intérêt du superquantile, en particulier dans

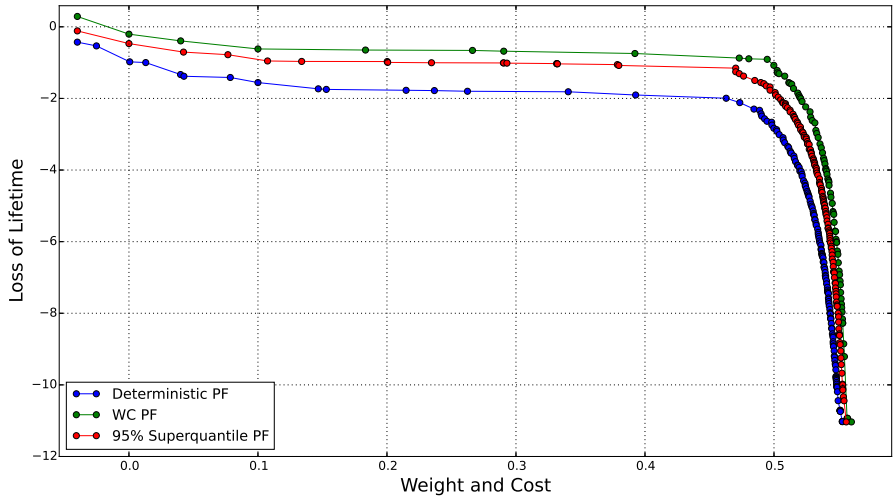


FIGURE 8.24 – Représentation dans l'espace des objectifs des points Pareto-optimaux pour la formulation déterministe, pour la formulation avec un pire cas et pour la formulation avec un 95%-superquantile pour quantifier les effets des incertitudes en entrée.

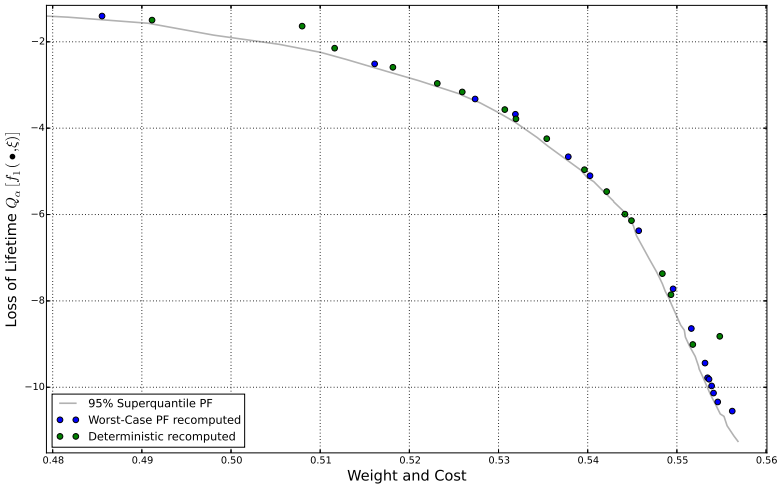


FIGURE 8.25 – Représentation du $Q_\alpha\left[f_1\left(\boldsymbol{\zeta}\left(\boldsymbol{x},\boldsymbol{\xi}_x\right),\boldsymbol{\xi}_e\right)\right]$ en fonction de f_2 des points Pareto-optimaux pour la formulation déterministe et pour la formulation avec un pire cas pour quantifier les effets des incertitudes en entrée. Zoom sur les solutions de type *rupture technologique*.

la zone où les solutions sont les plus innovantes, proposant une rupture technologique avec l'équipement de référence de coordonnées $(0,0)$ dans l'espace des objectifs.

Enfin, en comparant les coordonnées des paramètres de contrôle $\boldsymbol{x}$ entre les deux points calculés pour la figure 8.26 nous pouvons remarquer que les coordonnées de ces deux points sont relativement proches. Nous pouvons toutefois souligner que la solution pire cas a pour différence principale d'utiliser davantage de caloducs (R_{th} plus petite) en diminuant la ventilation ($\dot{m}$ plus petit) par rapport à la solution basée sur le superquantile.

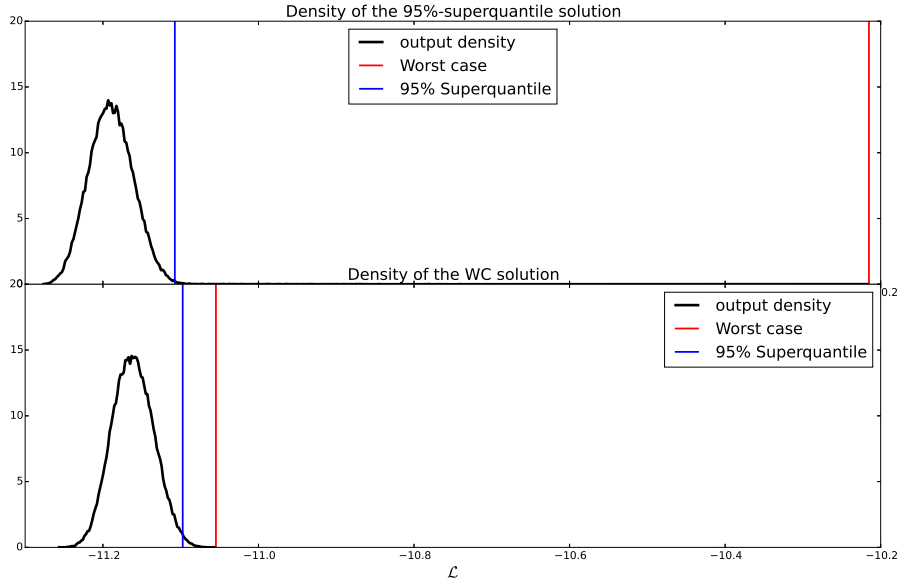


FIGURE 8.26 – Densité de la sortie de f_1 du point extrémal minimisant la perte de durée de vie f_1 du front de Pareto Pire cas de la figure 7.9 et du front de Pareto superquantile de la figure 8.22

8.5 Conclusions sur l'optimisation probabiliste par MOEGO NSGA-II couplé à l'algorithme d'entropie croisée

Dans cette partie nous avons proposé un algorithme de résolution de problèmes multi-objectifs avec un objectif incertain pris en compte par une mesure de risque probabiliste de type superquantile. Pour cela :

- Nous avons prouvé la continuité de la fonction $g_\rho(\boldsymbol{x}) = \rho_\xi \left[f(\boldsymbol{x}, \boldsymbol{\xi})\right]$ au théorème 8.1 afin de pouvoir appliquer l'algorithme MOEGO NSGA-II développé dans la section 6.4.
- Nous avons développé un algorithme d'estimation de superquantile dont les conclusions sont présentées en section 8.2.4. Ce dernier est ensuite utilisé pour estimer la fonction g_ρ en chaque point de contrôle $\boldsymbol{x}$ du problème multi-objectif. MOEGO NSGA-II et l'algorithme d'entropie croisée sont de plus couplés à l'aide de l'algorithme 7.1. Cette réutilisation a de nouveau fait preuve de son efficacité, sur le cas analytique et industriel.
- Nous avons montré que même si l'estimation du superquantile par l'algorithme 8.1 était bruitée, la possibilité d'utiliser la population finale retournée par NSGA-II appliqué sur les prédictions par krigeage était une alternative intéressante. En particulier si l'on connaît l'ordre de grandeur du bruit introduit, ceci permettant de fixer l'effet pépite du krigeage.
- Le temps de restitution de l'algorithme multi-objectif basé sur un superquantile est de l'ordre de 2000 appels à la fonction avec entrées incertaines f_1. Pour rappel, les cas tests utilisés pour déduire ce nombre d'appels ont quatre variables de contrôle et huit variables incertaines, dont les quatre de contrôle.
- Enfin, nous avons confirmé l'utilité sur le cas industriel et sur le cas analytique de

la prise en compte des incertitudes de manière probabiliste plutôt que déterministe par le pire cas. L'utilisation d'une mesure de risque probabiliste permet de proposer des solutions dont la robustesse est quantifiée (par le α du superquantile) et qui peuvent être plus performantes que celle proposée par la résolution du pire cas.

Toutefois, ce chapitre a également fait ressortir une limite importante qui est le choix des paramètres de la méthode d'entropie croisée. Nous avons en particulier remarqué que, selon le choix de ces paramètres, l'influence du bruit pouvait être pénalisante pour l'algorithme multi-objectif, qui doit de ce fait être utilisé différemment. L'estimateur de la variance asymptotique de l'erreur commise par l'estimation par échantillonnage préférentiel sur le superquantile fourni par le théorème 4.2 peut être une piste. En effet, ce dernier permet de déterminer un critère de décision : si la variance approchée calculée à la fin de l'algorithme 8.1 n'est pas satisfaisante, nous pouvons augmenter la taille N_{CE} pour l'optimisation. Également, elle peut permettre de déterminer l'effet pépite à intégrer à la construction du modèle de krigeage pour MOEGO NSGA-II.

Chapitre 9

Construction de réseaux de neurones récurrents pour la prédiction de phénomènes spatio-temporels

Sommaire

Les simulations thermiques transitoires permettant de calculer la fonction objectif f_1 du problème de l'équation (2.12) peuvent être coûteuses et nécessiter plusieurs heures de calcul. Or, les réductions de modèles physiques existantes ont pour désavantage d'être incapables de prendre en compte les non-linéarités des phénomènes physiques telles que le passage de la convection naturelle à la convection forcée, comme cela est expliqué à la section 2.2.1.2. De plus, comme les conclusions des sections 7.3 et 8.5 le relèvent, même avec

des techniques les plus économiques possibles, l'optimisation sous incertitudes à l'aide de mesures de risque est gourmande en nombre d'appels à la fonction d'estimation de la durée de vie. L'objectif de cette partie est donc de développer un algorithme de construction de modèles de substitution spatio-temporels n'utilisant qu'un nombre limité de trajectoires provenant du modèle de référence, avec des dimensions d'entrées-sorties qui peuvent être importantes (puisqu'il peut y avoir beaucoup de zones à connaître), et capable de prédire des phénomènes de plusieurs milliers de seconde, du type d'un profil de vol. La section 9.1 justifie ce besoin et explique le cadre. Nous proposons d'abord de partir de l'état de l'art, dont nous avons fait la revue à la section 3.3, et d'en comprendre les lacunes. Ensuite, nous proposons de développer une méthodologie de construction de réseaux de neurones récurrents adaptée au cadre de la section 9.1 : la section 9.2 répond au problème du fléau de la dimension auquel peut faire face un réseau rebouclé. La section 9.3 permet d'adapter au cas spatio-temporel des techniques d'apprentissage statistique répandues dans le cas stationnaire mais dont l'usage dans un cadre spatio-temporel n'est pas immédiat. En particulier, les questions de la sélection de modèle et de la construction de plans d'expériences sont discutées. Enfin, cette méthodologie est testée sur un cas de panne de l'extraction de l'air d'un modèle d'équipement électronique.

9.1 Description du cas test thermique - Cadre pour la construction de réseaux neuronaux

9.1.1 Besoin en ingénierie thermique

9.1.1.1 Cadre de la réduction de modèle thermique

Le calcul de la durée de vie d'un équipement électronique détaillé à la section 2.2.2 nécessite l'exécution de simulations thermiques instationnaires, c'est-à-dire l'estimation de la température de différentes zones d'un équipement lorsque ce dernier est soumis à des conditions aux limites dépendantes du temps. En particulier, la température des composants électroniques critiques lors d'un profil de vol type est la quantité à connaître pour déduire la durée de vie comme expliqué dans la section 2.2.2.

Pour cela, il s'avère nécessaire d'utiliser des simulations numériques coûteuses en temps de calcul par le biais de modèles détaillés, comme décrits en section 2.2.1. En particulier, ceci est le cas lorsque l'une des conditions aux limites variant au cours du temps est le débit d'extraction de l'air de l'équipement. En effet, si au cours d'un vol l'extraction de l'air ne fonctionne plus, le régime convectif à l'intérieur de l'équipement passe de la convection forcée à la convection naturelle. Or, cette transition est source de fortes non-linéarités comme expliqué à la section 2.2.1.1. Ceci rend les réductions de modèle physiques existantes inadaptées, comme le détaille la section 2.2.1.2. C'est la raison pour laquelle la construction de modèles de substitution par apprentissage statistique, introduite en section 3.3, représente une alternative intéressante.

Pour rappel et en reprenant les notations de cette section, l'objectif de l'apprentissage statistique pour la construction de modèles de substitution spatio-temporels est de construire un modèle d'exécution rapide $\hat{f}$ permettant de prédire l'évolution des sorties $\boldsymbol{y}^k := \left(y_1(t^k), \cdots, y_{N_y}(t^k)\right)\ \forall k \geqslant 0$ uniquement à partir de la connaissance de la valeur initiale de ces sorties : $\boldsymbol{y}^0 := \boldsymbol{y}(t^0)$ et des entrées exogènes jusqu'au temps t^k : $\boldsymbol{u}^0 := \boldsymbol{u}(t^0)$, $\cdots$, $\boldsymbol{u}^k := \boldsymbol{u}(t^k)$. Les valeurs prédites par le modèle de substitution sont écrites $\hat{\boldsymbol{y}}^k$ afin de

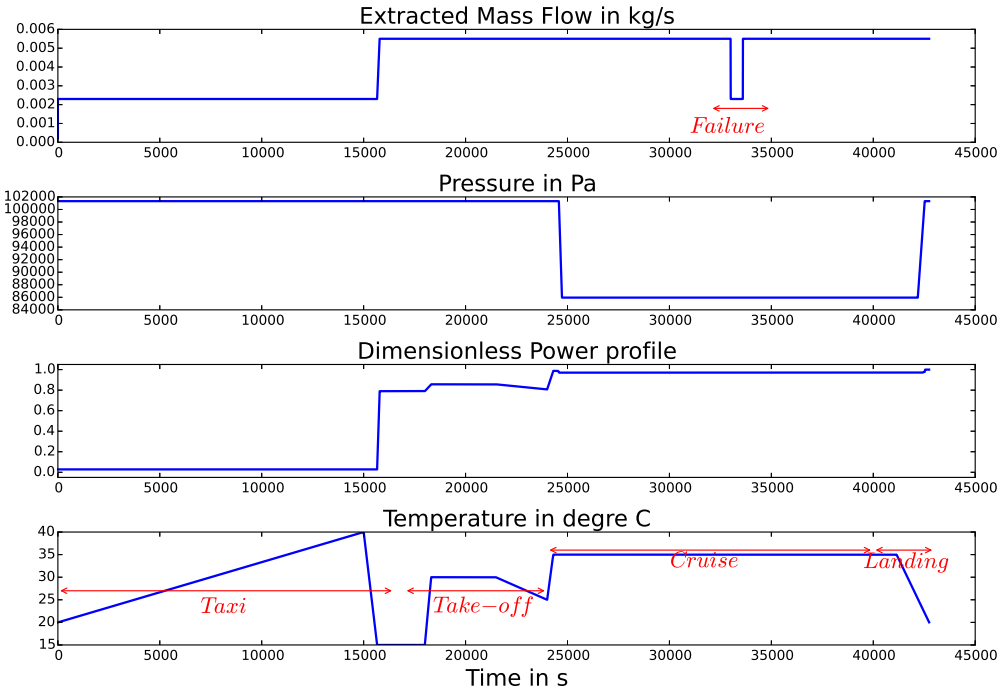


FIGURE 9.1 – Profil de vol type avec stationnement en zone tempérée, atterrissage en zone tempérée et panne de l'extraction de l'air de l'équipement pendant 10 minutes.

les différencier des valeurs mesurées, c'est-à-dire provenant du système à prédire, $\boldsymbol{y}^k$. Ici, ces sorties proviennent d'un modèle numérique coûteux. Ceci implique que le plan d'expériences, défini à l'équation (3.34) en spatio-temporel, est de taille réduite. En pratique, nous ne pouvons utiliser qu'une trentaine de trajectoires d'apprentissage.

La construction d'un modèle de substitution spatio-temporel passe par plusieurs étapes. Par exemple, il est nécessaire de choisir un modèle-hypothèse qui convient. De cela, on peut ensuite en déduire la manière d'utiliser le modèle de substitution $\hat{f}$, comme expliqué en section 3.3.1. Dans notre cas, les sorties provenant du modèle de référence sont considérées non bruitées, nous pouvons donc utiliser $\hat{f}$ en simulateur d'après le tableau 3.1. Ceci signifie pour $\hat{f}$ que sa dimension d'entrée peut être importante puisqu'elle contient les entrées exogènes **et** les sorties rebouclées, comme illustré en figure 3.16.

9.1.1.2 Cas test : prédiction d'un cas de panne de l'extraction de l'air au cours d'un vol

Le cas test étudié représente un équipement électronique comme détaillé dans la partie 2.3. La différence principale de ce chapitre réside dans les profils de vol utilisés. Afin de se placer dans un cas défavorable aux réductions de modèle physiques existantes, le profil de vol utilisé n'est plus le même que pour les chapitres précédents. Pour cela, nous choisissons un profil de vol contenant une panne de l'extraction de l'air. Les autres conditions aux limites variant en temps sont la pression et la température ambiante de la cavité dans laquelle se trouve l'équipement et l'intensité du courant le traversant. Le profil de test de ces conditions aux limites est donné sur la figure 9.1. Sa durée est de 42740 s.

En plus de ces quatre entrées exogènes, le cas test présente six températures d'intérêt

à prédire dans l'équipement et détaillées sur la figure 9.2.

1. Température de la paroi supérieure.
2. Température de la paroi opposée au composant critique.
3. Température du radiateur lié au composant.
4. Température de l'air interne de l'équipement.
5. Température du composant électrique.
6. Température de la paroi sur laquelle est placée le composant critique.

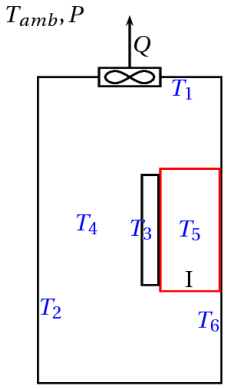


FIGURE 9.2 – Représentation schématique de l'équipement modélisé.

En ce qui concerne les trajectoires d'apprentissage, nous ne pouvons nous permettre d'utiliser des trajectoires aussi longues que le profil de vol puisque le temps de construction ne doit pas être trop important. Nous nous limitons donc à 37 trajectoires de 300 s. La génération de ces trajectoires est réalisée aléatoirement. Ceci est expliquée en section 9.3.2.

9.1.2 Cadre pour la construction de réseaux neuronaux

Avant de déterminer le modèle analytique pour $\hat{f}$, résumons le cadre de construction de ce modèle de substitution spatio-temporel :

1. Il y a peu de trajectoires d'apprentissage disponibles. Du fait du coût élevé d'exécution des simulations thermiques instationnaires de référence, la taille du DOE est réduite (de l'ordre de 30 trajectoires). Toutefois, chacune de ces trajectoires est discrétisée temporellement, avec un pas de temps constant Le nombre de points d'apprentissage pour le modèle $\hat{f}$ peut ainsi facilement être de l'ordre de plusieurs milliers.
2. La dimension d'entrée de $\hat{f}$ peut être élevée. Puisque le modèle $\hat{f}$ est utilisé en simulateur, toutes les sorties du modèle $\hat{f}$ deviennent les entrées pour prédire la sortie au temps suivant, comme illustré sur la figure 9.3. De ce fait, le nombre d'entrées de $\hat{f}$ peut rapidement devenir important. Par exemple, une dimension d'entrée pour $\hat{f}$ de 10 est facilement atteinte.
3. Les phénomènes à prédire sont non-linéaires et se déroulent sur des temps longs. L'objectif est d'estimer la température de l'équipement au cours d'un vol. Cela signifie que, pour l'application, les profils temporels des entrées exogènes peuvent aisément atteindre les 50 000 s avec un pas de temps de l'ordre de la seconde. En ce qui concerne la non-linéarité, elle est justifiée à la section 2.2.1.1. Il y est mis en évidence que la physique à prédire est non-linéaire. Ces non-linéarités peuvent provenir du rayonnement thermique, du changement de régime convectif ou encore de la puissance dissipée par un composant électronique par effet Joule ($\Phi = R(T)I$) puisque la résistance électrique peut dépendre de la température.

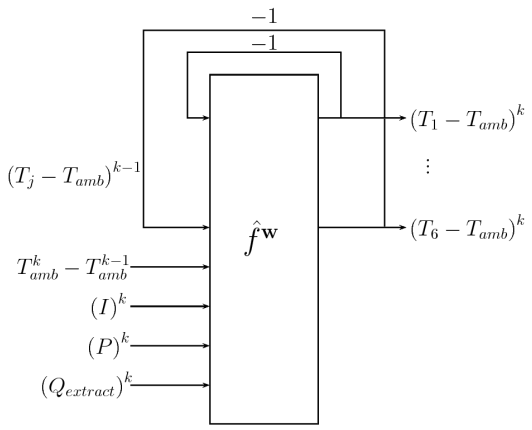


FIGURE 9.3 – Illustration du simulateur à un pas du comportement thermique d'un équipement électronique à construire.

4. Le temps de construction doit être raisonnable et le temps d'exécution du modèle de substitution doit être négligeable. En effet, pour que la réduction de modèle ait un intérêt, il ne faut pas que la construction du métamodèle soit plus lourde que l'utilisation du modèle de référence. Cela dépend néanmoins de la manière dont ces simulations doivent être utilisés. Par exemple, si l'objectif est de l'utiliser pour calculer une fonction objectif en optimisation, cela peut requérir un nombre d'appels pouvant atteindre le millier. Un modèle de substitution de ce type peut donc rapidement devenir rentable.

9.1.3 Choix du modèle analytique $\hat{f}$ et construction existante

Pour reprendre la démarche d'identification de système détaillée à la figure 3.14, il est à présent nécessaire de choisir l'ordre en temps du modèle-hypothèse et donc du simulateur. Comme le fait De Lozzo dans son étude [Lozzo *et al.*, 2013], que nous détaillons à la section 3.3.1.3, nous pouvons utiliser l'information fournie par la physique que nous souhaitons prédire : notamment, nous savons que la thermique est une physique régie par des équations du premier ordre en temps (d'après la section 2.2.1.1). On peut donc en déduire que, si la discrétisation spatiale du domaine est bien effectuée, un modèle-hypothèse d'ordre 1 en temps comme écrit à l'équation (3.33) est un choix adapté. En ce qui concerne la discrétisation spatiale, elle peut être réalisée comme pour la réduction de modèle physique expliquée en section 2.2.1.2. Ceci signifie que c'est par l'expertise d'un thermicien que le choix des températures à prédire et à reboucler doit être fait puisqu'il est nécessaire de ne pas perdre d'information sur les chemins qu'emprunte la chaleur pour être évacuée de l'équipement. Il reste à choisir le modèle analytique pour $\hat{f}$. Comme nous l'avons précisé au point 3 de la section 9.1.2, les phénomènes à prédire peuvent être fortement non-linéaires, ce qui exclut donc les régressions linéaires. De plus, le point 1 de la section 9.1.2 précise que le nombre de points d'apprentissage pour le modèle $\hat{f}$ peut aisément atteindre plusieurs milliers (issus de seulement quelques trajectoires) et le point 4 que l'exécution du modèle $\hat{f}$ doit être pratiquement instantanée. Ceci permet d'exclure les modèles de krigeage dont l'exécution s'alourdit avec le nombre d'échantillons dans le DOE. Le calcul de la prédiction par krigeage requiert l'inversion d'une matrice carrée $\mathbf{\Gamma}$ (définie à l'équation (3.21)) de la

taille du nombre d'échantillons dans le DOE, ce qui explique que le temps d'exécution en dépende fortement.

Pour construire un modèle dont le temps d'exécution reste négligeable malgré le nombre de points discrétisés du DOE, il faut faire appels à des modèles régressants du type des réseaux de neurones définis à la section 3.2.2. Ils ont pour avantage d'être parcimonieux, c'est-à-dire de nécessiter l'utilisation d'un nombre réduit de neurones afin de prédire des phénomènes qui peuvent être fortement non-linéaires. De plus, ils ont pour propriété d'être des approximateurs universels, donc que toute fonction (linéaire ou non) peut être prédite avec la précision souhaitée [Cybenko, 1989] [Barron, 1993] en jouant sur la complexité du modèle de substitution. Enfin, des réseaux de neurones récurrents pour la construction de modèles de substitution spatio-temporels ont déjà été mis en œuvre dans le cadre de l'ingénierie thermique dans l'étude de De Lozzo [Lozzo *et al.*, 2013]. Ils sont donc particulièrement indiqués dans notre cas.

Intéressons nous donc à leur mise en place dans un cadre spatio-temporel. Pour résumer l'état de l'art du chapitre 3, la construction d'un réseau de neurones pour la prédiction spatio-temporelle se décompose en plusieurs étapes :

- **Initialisation des poids $\boldsymbol{w}$** à l'aide de la méthode proposée par Nguyen-Widrow dans leur étude [Nguyen et Widrow, 1990]. Elle consiste à initialiser aléatoirement les poids d'un réseau à une couche cachée. Les lois de probabilité utilisées sont différentes pour les poids $\boldsymbol{w}'$ liant la couche d'entrée et la couche cachée et ceux, $\boldsymbol{w}''$, liant la couche cachée et la couche de sortie.
- **Optimisation des poids $\boldsymbol{w}$** par minimisation de l'erreur quadratique moyenne (définie à l'équation (3.2) avec l'erreur définie à l'équation (3.35) en spatio-temporel) commise par le réseau récurrent en chaque temps des trajectoires d'apprentissage du DOE. Le gradient de cette erreur se calcule efficacement par BPTT détaillé dans [Werbos, 1990] et basé sur la rétropropagation de la section 3.2.2.2. Des techniques d'optimisation par gradient, telles que Levenberg-Marquardt, sont donc utilisées pour résoudre ce problème d'optimisation. L'apprentissage d'un réseau de neurones est détaillé dans la section 3.2.2.1.
- **Sélection du modèle**, c'est-à-dire du nombre de neurones optimal et des poids permettant de réaliser le compromis entre biais et variance du modèle illustré à la section 3.1. Pour cela, la validation croisée définie à l'algorithme 3.2 est mise en place. La section 3.3.2.2 expose différentes manières de répartir des trajectoires temporelles dans les différents *folds*.

Toutefois, cette construction est inadaptée au cadre d'application visé et donné dans la section 9.1.2. En premier lieu, le point 2 du cadre de la section 9.1.2 peut être un obstacle à la résolution du problème d'optimisation permettant de calculer les poids. Par exemple, pour la construction d'un modèle de substitution du cas test de la section de la section 9.1.1.2, $\hat{f}$ peut avoir 10 entrées, ce qui n'est pas encore considéré comme de la dimension élevée. Malgré cela, la dimension des paramètres de contrôle du problème d'optimisation de l'équation (3.2) peut facilement approcher les 200. Par exemple avec un nombre de neurones réduit à $C = 11$ et avec $N_y = 6$ sorties et $N_x = 10$ entrées, on obtient 196 poids à calculer d'après le calcul de l'équation (3.13). Or, plus la dimension d'entrée d'un problème d'optimisation est importante, plus ce dernier est difficile à résoudre. En particulier, puisque des algorithmes d'optimisation locaux sont utilisés, plus la dimension est élevée, plus il est probable que de nombreux optima locaux existent, complexifiant ainsi le calcul des poids. La section 9.2 introduit deux méthodes permettant d'adapter la

construction à ces contraintes.

En second lieu, la sélection de modèle proposée ici nécessite de scinder le DOE en 5 *folds* différents. La manière proposée par De Lozzo [Lozzo *et al.*, 2013] et expliquée à la figure 3.19 présente l'avantage de répartir l'information provenant de toutes les trajectoires dans chacun des *folds*, à la différence de la répartition des trajectoires entières. Néanmoins, cette répartition par *sous-trajectoires* a un inconvénient majeur. À chaque étape de l'algorithme de validation croisée et du fait de la répartition de points proches les uns des autres dans les différents sous-ensembles, le *fold* de test a une grande proximité avec le sous-ensemble d'apprentissage. L'erreur sur le sous-ensemble de test se comporte donc comme une erreur d'apprentissage. Comme expliqué dans la section 3.1, l'erreur d'apprentissage décroît donc avec la complexité et de ce fait elle ne permet pas de trouver le nombre de neurones qui réalise le compromis entre biais et variance du modèle. La section 9.3 propose une solution à ce problème.

9.2 Adaptation des réseaux de neurones récurrents à la montée en dimension

Pour étendre la construction de réseaux de neurones récurrents proposée par De Lozzo [Lozzo *et al.*, 2013] à une dimension d'entrée élevée, nous proposons d'abord de modifier la construction et plus précisément la phase d'optimisation des poids du réseau. Plus précisément, puisque la dimension des poids à calculer est considérée comme importante, nous décomposons l'optimisation de l'équation (3.2) en sous-problèmes de dimension réduite et donc plus faciles à résoudre. Puis, nous mettons en place une technique d'analyse de sensibilité de sorte à pouvoir diminuer la dimension de chacun des sous-problèmes précédents.

9.2.1 Optimisation multi-niveaux des poids du réseau

L'un des principaux désavantages de la construction de réseaux de neurones en présence d'un nombre d'entrées et de sorties important est que les poids optimaux sont la solution d'un problème d'optimisation de grande dimension. Or, plus la dimension est importante et plus le problème d'optimisation à résoudre est compliqué et plus les solutions obtenues ont de chance de ne pas être optimales.

Il est donc préférable de résoudre des problèmes d'optimisation de dimension inférieure. Nous pouvons pour cela décomposer la résolution de l'optimisation de l'équation (3.2) en sous-problèmes de dimension plus réduite, et donc plus faciles à résoudre. Une mise en œuvre intuitive est de minimiser l'erreur commise par le réseau de neurones sortie par sortie, et donc de ne jouer que sur les poids liés à chacune des sorties séparément. En effet, cela permet de ne calculer que les poids permettant de calculer la sortie correspondante à chaque fois. Nous devons alors introduire une nouvelle notation pour les poids du réseau afin de les décomposer selon la sortie qu'ils permettent de calculer.

$$\left\{ \begin{array}{rcl} \boldsymbol{w} & = & (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_{N_y}) \\ \boldsymbol{w}_j & = & \left(\boldsymbol{w}'_{i,\mathcal{N}_j}, \boldsymbol{w}''_{\mathcal{N}_j,j}\right)_{1 \leqslant i \leqslant N_y + N_u} \quad \text{pour } 1 \leqslant j \leqslant N_y \end{array} \right. \tag{9.1}$$

Soit $\mathcal{N}_j$ les neurones du sous-réseau j :

$$\mathcal{N}_j = \{n \in [\![1, C]\!] \ : \ \text{neurone } n \in j^{eme} \text{ sous} - \text{réseau}\} \tag{9.2}$$

Cette notion de sous-réseau permet d'associer des poids $\boldsymbol{w}_j$ et un nombre de neurones C_j à chaque sortie j. En effet, puisque nous souhaitons minimiser l'erreur séparément pour chaque sortie, le nombre de neurones total du réseau complet C est choisi pour chaque sortie indépendamment. Le sous-réseau permettant de prédire la sortie $j \in [\![1, N_y]\!]$ est dès lors également défini par les C_j neurones qui le compose. Ainsi, le réseau total a pour nombre de neurones $C = \sum_{j=1}^{N_y} C_j$. L'équation (9.2) permet en fait de définir l'ensemble d'indices $\mathcal{N}_j$ des neurones appartenant au $j^{ème}$ sous-réseau. Ces sous-réseaux sont représentés à la figure 9.4.

Pour résumer, nous cherchons les $\boldsymbol{w}_j$ qui minimisent l'erreur commise par la sortie du j^{eme} ($1 \leqslant j \leqslant N_y$) sous-réseau $\hat{f}_j$ (le réseau entier étant défini à l'équation (3.12)).

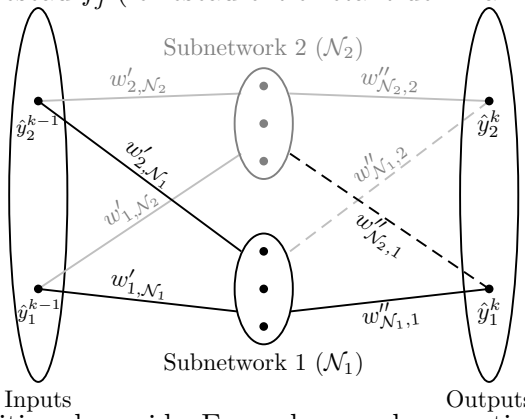


FIGURE 9.4 – Décomposition des poids. Exemple avec deux sorties $\hat{y}_1$ et $\hat{y}_2$. En pointillés sont représentés les poids non utilisés à l'itération 0 détaillée à l'algorithme 9.1.

Maintenant que nous avons introduit les notations permettant de décomposer le problème d'optimisation en sous-problèmes, nous devons en expliquer la résolution et le lien avec le problème initial. Pour rappel, ce dernier est celui de l'équation (3.2) avec l'erreur définie à l'équation (3.35). Afin de justifier la décomposition choisie, réécrivons cette erreur :

$$\begin{aligned} \mathcal{E}(\hat{f}\,(\bullet;\ \boldsymbol{w})\,, DOE) &= \sum_{i=1}^{N_{DOE}} \sum_{k=1}^{N_t} \sum_{j=1}^{N_y} \left(y_j^{k,i} - \hat{f}_j(\boldsymbol{u}^{k-1,i}, \hat{\boldsymbol{y}}^{k-1,i};\ \boldsymbol{w})\right)^2 \\ &= \sum_{j=1}^{N_y} \mathcal{E}(\hat{f}_j\,(\bullet;\ \boldsymbol{w})\,, DOE) \end{aligned} \tag{9.3}$$

$\hat{f}_j$ correspond à la j^{ieme} sortie du réseau de neurones entier $\hat{f}$. Or, l'erreur de chacune de ces sorties s'écrit de la manière suivante :

$$\begin{aligned} \mathcal{E}(\hat{f}_j\,(\bullet;\ \boldsymbol{w})\,, DOE) &= \sum_{i=1}^{N_{DOE}} \sum_{k=1}^{N_t} \left(y_j^{k,i} - \hat{f}_j(\boldsymbol{u}^{k-1,i}, \hat{y}_1^{k-1,i}, \cdots, \hat{y}_j^{k-1,i}, \cdots, \hat{y}_{N_y}^{k-1,i};\ \boldsymbol{w})\right)^2 \\ &= \sum_{i=1}^{N_{DOE}} \sum_{k=1}^{N_t} \left(y_j^{k,i} - \hat{f}_j(\boldsymbol{u}^{k-1,i}, \hat{y}_j^{k-1,i}, \cdots, \hat{\boldsymbol{y}}_{i\neq j}^{k-1,i};\ \boldsymbol{w})\right)^2 \end{aligned} \tag{9.4}$$

On remarque donc que pour trouver les poids $\boldsymbol{w}_j$ qui minimisent l'erreur de l'équation (9.4) de chacun des réseaux séparément, il est nécessaire d'avoir accès à la prédiction par le réseau des autres sorties $\hat{\boldsymbol{y}}_{i\neq j}^{k,l} = \hat{f}_{i\neq j}(\boldsymbol{u}^{k-1,i}, \hat{\boldsymbol{y}}^{k-1,i};\ \boldsymbol{w})$. Ceci implique que les

poids permettant de calculer les autres sorties doivent également être connus. Or, ils ne sont pas accessibles au début du processus et il est nécessaire de mettre en place une optimisation multi-niveaux et donc itérative : à chaque itération it, les poids calculés à l'itération précédente $\boldsymbol{w}_{i\neq j}^{it-1}$ sont utilisés afin de pouvoir reboucler les sorties en entrée du sous-réseau j duquel nous calculons les poids. De plus au début du processus, ni les poids ni le nombre de neurones optimal des autres sous-réseaux ne sont connus. Il est donc nécessaire d'utiliser les valeurs mesurées, donc provenant du modèle de référence, des sorties $i \neq j$. Pour rappel, elles sont notées sans chapeau donc $\boldsymbol{y}_{i\neq j}^{k,l}$. Ce qui signifie qu'initialement les poids sont calculés en résolvant l'équation suivante :

$$\begin{array}{l} \min\limits_{\boldsymbol{w}_j^0} \sum\limits_{i=1}^{N_{DOE}} \sum\limits_{k=1}^{N_t} \left(y_j^{k,i} - \hat{f}_j(\boldsymbol{u}^{k-1,i}, \hat{y}_j^{k-1,i}, \cdots, \boldsymbol{y}_{i\neq j}^{k-1,i};\ \boldsymbol{w}) \right)^2 \\ \text{avec } \left\{ \begin{array}{rcl} \boldsymbol{w}_j & = & \left(\boldsymbol{w}_j^0, \boldsymbol{w}_{N_i \neq j, j} = 0 \right) \\ \boldsymbol{w} & = & (0, \cdots, \boldsymbol{w}_j, \cdots, 0) \end{array} \right. \end{array} \tag{9.5}$$

L'équation (9.5) correspond à la minimisation des poids que nous devons mettre en place pour initialiser l'optimisation multi-niveaux. Toutefois, cette construction itérative peut être faite de deux façons :

1. La première est d'actualiser séquentiellement les poids des sous-réseaux $\boldsymbol{w}_j$ au fur et à mesure qu'ils sont calculés en commençant par la première sortie. Par exemple, on calcule d'abord les poids de la première sortie $\boldsymbol{w}_1$ solution de l'équation (9.5), puis les poids du deuxième sous-réseau sont calculés en utilisant les $\boldsymbol{w}_1$ précédemment calculé en résolvant le problème suivant :

$$\begin{array}{l} \min\limits_{\boldsymbol{w}_2} \sum\limits_{i=1}^{N_{DOE}} \sum\limits_{k=1}^{N_t} \left(y_2^{k,i} - \hat{f}_j(\boldsymbol{u}^{k-1,i}, \hat{y}_1^{k-1,i}, \hat{y}_2^{k-1,i}, \cdots, \boldsymbol{y}_{i\neq 1, i\neq 2}^{k-1,i};\ \boldsymbol{w}) \right)^2 \\ \text{avec } \boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2, 0, \cdots, 0) \end{array}$$

   Et ainsi de suite, jusqu'à ce que tous les poids soient calculés.

2. La seconde consiste à entièrement distribuer la construction des sous-réseaux et de ce fait de n'utiliser pour la phase d'initialisation que des entrées mesurées à la place des autres sorties rebouclées. En d'autres termes, elle consiste à ne résoudre que des problèmes du type de l'équation (9.5).

De manière à réduire le temps de construction tout en profitant de cette décomposition, nous faisons le choix d'entièrement distribuer la construction des sous-réseaux et donc d'utiliser la seconde procédure. Pour cela, il est nécessaire de différentier la phase d'initialisation, où nous calculons le nombre de neurones optimal par validation croisée, et la phase itérative, où le nombre de neurones est fixé et où l'on prend en compte le fait que les entrées sont prédites et non mesurées.

Ces deux phases sont résumées dans les algorithmes 9.1 et 9.2.

L'algorithme 9.1 permet d'initialiser les poids de sorte à pouvoir, aux itérations suivantes, utiliser les sorties rebouclées $\hat{\boldsymbol{y}}_{i\neq j}$ en entrée du sous-réseau j pour calculer les poids

Algorithme 9.1 Initialisation de l'optimisation multi-niveaux des poids

· Initialisation des poids $\boldsymbol{w}^0$ par Nguyen-Widrow [Nguyen et Widrow, 1990] avec $\boldsymbol{w}_j^0 = \left(\boldsymbol{w}'_{i,\mathcal{N}_j}, \boldsymbol{w}''_{\mathcal{N}_j,j}\right)_{1 \leqslant i \leqslant N_y+N_u}$;
for $j \in [\![1, N_y]\!]$ **do**
  · Calcul des poids $\boldsymbol{w}_j^{0\star}$ en résolvant l'équation (9.5);
  · Choisir la complexité optimale C_j du sous-réseau $\mathcal{N}_j$ par validation croisée de l'algorithme 3.2;
end
· Mettre à jour $\boldsymbol{w}^0 = \left(\boldsymbol{w}_1^{0\star}, \cdots, \boldsymbol{w}_{N_y}^{0\star}\right)$;

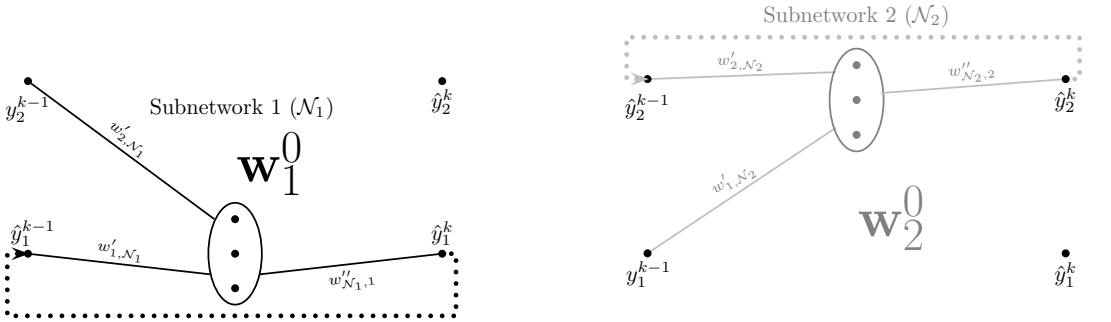


FIGURE 9.5 – Illustration des sous-réseaux optimisés de manière distribuée à l'itération 0 et détaillée à l'algorithme 9.1. Exemple avec deux sorties et en omettant les entrées exogènes.

$\boldsymbol{w}_j$. Aux itérations $it > 0$, l'équation à résoudre devient :

$$\begin{array}{l} \min\limits_{\boldsymbol{w}_j^{it}} \left\{ \sum\limits_{l=1}^{N_l} \sum\limits_{k=1}^{N_t} \left(y_j^{k,l} - \hat{f}_j \left(\boldsymbol{u}^{k-1,i}, \hat{y}_j^{k-1,l}, \hat{\boldsymbol{y}}_{i \neq j}^{k-1,l}; \boldsymbol{w} \right) \right)^2 \right\} \\ \text{avec} \left\{ \begin{array}{rcl} \boldsymbol{w} & = & \left(\boldsymbol{w}_1^{it-1}, \cdots, \boldsymbol{w}_j^{it}, \cdots, \boldsymbol{w}_{N_j}^{it-1} \right) \\ \boldsymbol{w}_j^{it} & = & \left(\boldsymbol{w}'_{i_1,\mathcal{N}_j}, \boldsymbol{w}''_{\mathcal{N}_j,j}, \boldsymbol{w}''_{\mathcal{N}_{i_2 \neq j},j} \right)^{it}_{1 \leqslant i_1 \leqslant N_y+N_u} \end{array} \right. \end{array} \tag{9.6}$$

Par ailleurs, il est important de noter que lors de cette étape d'initialisation, les poids liant le sous-réseau j aux sorties $i \neq j$ ne sont pas calculés. Ces poids, illustrés par les lignes en pointillés sur la figure 9.4, sont une des raisons pour lesquelles il est important d'itérer dans l'algorithme 9.2. Ces itérations sont en effet nécessaires pour calculer tous les poids du réseau.

Enfin, afin de choisir si les poids calculés à l'itération it doivent être mis à jour ou non, l'erreur de validation croisée à l'itération it doit être inférieure à celle obtenue aux itérations précédentes. En ce qui concerne le critère d'arrêt, dès qu'il existe une itération durant laquelle l'erreur de validation croisée ne décroît plus pour toutes les sorties, l'algorithme s'arrête naturellement puisqu'il n'y a plus aucune chance que cette erreur décroisse à nouveau : l'algorithme d'optimisation utilisé étant déterministe, les poids retournés sont toujours les mêmes et donc il en va de même pour l'erreur de validation croisée.

Pour résumer les étapes de l'optimisation multi-niveaux mise en place, elle est décomposée en deux étapes. La première étape consiste à initialiser les poids des sous-réseaux en utilisant des entrées mesurées comme illustré sur la figure 9.5. Puis, afin de prendre en compte dans le calcul des poids optimaux le fait que les entrées correspondent à des

Algorithme 9.2 Optimisation multi-niveaux des poids

· $it = 0$;
· Calculer $\boldsymbol{w}^0$ à l'aide de l'algorithme 9.1;
· Récupérer l'erreur de validation croisée pour chacune des sorties → CV^0_{error} 9.1;
· Poser $StoppingCrit = False$ et $\mathrm{CV}^{min}_{error} = \mathrm{CV}^0_{error}$;
while $StoppingCrit == False$ **do**
  · $it = it + 1$;
  · $StoppingCrit = True$;
  for $j \in [\![1, N_y]\!]$ **do**
    · calculer $\boldsymbol{w}_j^{it\star}$ solution de l'équation (9.6);
    if $\mathrm{CV}^{min}_{error\,j} \geqslant \mathrm{CV}^{it}_{error\,j}$ **then**
      · $StoppingCrit = False$;
    end
  end
end
· $\boldsymbol{w} = \boldsymbol{w}^{it\star}$;

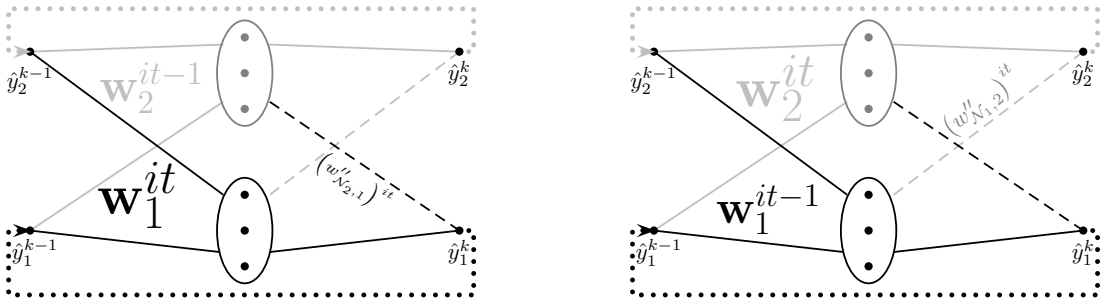


FIGURE 9.6 – Illustration des sous-réseaux optimisés de manière distribuée aux itérations $it > 0$ et détaillée à l'algorithme 9.2. Exemple avec deux sorties et en omettant les entrées exogènes.

sorties rebouclées et de sorte à calculer tous les poids, on utilise les poids de l'itération précédente pour calculer les sorties rebouclées et choisir les poids pour chaque sortie qui minimisent l'erreur de l'équation (9.6) (comme illustré sur la figure 9.6). L'algorithme se termine une fois que l'erreur de validation croisée ne décroît plus.

Enfin, puisque l'erreur de validation croisée est une composante fondamentale de cette optimisation des poids, il est impératif qu'elle fournisse une bonne estimation de l'erreur de généralisation. À cette fin, nous introduisons en section 9.3 une méthode de répartition des trajectoires dans les *folds*.

9.2.2 Application de l'analyse de sensibilité au modèle non rebouclé pour la réduction de dimension

L'analyse de sensibilité, détaillée à la section 4.3, est une technique qui peut être utilisée pour quantifier l'effet de la variation d'une entrée sur la sortie d'un modèle. Cela peut en particulier permettre de réduire la dimension d'entrée du réseau de neurones en détectant les entrées les moins informatives. De plus, réduire la dimension d'entrée du réseau sert également à réduire la dimension du problème d'optimisation à résoudre pour construire le réseau de neurones. Comme cela est détaillé dans la section 4.3.1, l'une des

techniques d'analyse de sensibilité globale la plus répandue est l'estimation des indices de Sobol, donné à l'équation (4.33). Elle consiste à évaluer la part de variance de la sortie expliquée par la variation de chacune des entrées.

Dans le cas où les entrées ne sont pas corrélées, la décomposition de Hoeffding (donnée à l'équation (4.31)) permet de donner du sens à ces coefficients puisque cela permet de montrer que la somme des indices de Sobol est égale à 1. De cette manière, chaque indice de Sobol représente un pourcentage de variance due à l'entrée étudiée.

En particulier, l'indice de Sobol total permet de connaître l'influence totale (incluant les interactions avec les autres entrées) d'une entrée sur la sortie. Cette caractéristique fait de cet indice le meilleur indicateur possible pour décider de supprimer une entrée. Néanmoins, son estimation peut être coûteuse. Une autre technique détaillée dans la section 4.3.1 appelée DGSM donne une alternative aux indices de Sobol dans le cas où le gradient de la sortie est connu. Ces coefficients sont donnés à l'équation (4.35). Si les entrées ne sont pas corrélées, ils sont une majoration à un coefficient près des indices de Sobol totaux.

De plus, dans le cas où le gradient peut être facilement calculable, les coefficients DGSM sont plus efficients à calculer : ils ne nécessitent que l'estimation d'une espérance. En particulier dans le cas des réseaux de neurones, la dérivée $\frac{\partial \hat{f}_j(\boldsymbol{x}\ ;\ \boldsymbol{w})}{\partial x_i}$ est connue analytiquement. Une méthode de Monte-Carlo, comme celle de l'étude [Kucherenko *et al.*, 2009] peut donc être appliquée à moindres frais pour estimer les coefficients DGSM.

À présent, il faut déterminer la manière dont ces coefficients peuvent être estimés sur notre série temporelle. Pour cela, nous utilisons le fait que $\hat{f}$ ne dépende pas du temps, c'est-à-dire que le temps n'est pas une entrée du réseau de neurones et les poids de ce dernier sont fixés une fois la construction terminée. Nous pouvons dès lors calculer les coefficients DGSM directement sur le modèle de substitution non rebouclé $\hat{f}$. Cela revient à ne réaliser l'analyse de sensibilité que sur le passage d'un pas de temps à un autre.

Les coefficients DGSM nécessitent de connaître le gradient de chaque sortie du réseau. Ce dernier est donné par la formule suivante :

$$\frac{\partial \hat{f}_j\left(\boldsymbol{x}\ ;\ \boldsymbol{w}\right)}{\partial x_i} = \sum_{n=1}^{C_j} w''_{n,j} w'_{i,n} \phi' \left(\sum_{p=1}^{N_e} w'_{p,n} x_p + w'_{0,n} \right) \tag{9.7}$$

x_i représente ici n'importe quelle entrée du modèle $\hat{f}$ non rebouclé : en ce sens, elle peut correspondre à une sortie rebouclée ou à une entrée exogène.

Pour rappel, dans notre cas les entrées sont corrélées. Ceci empêche l'utilisation de la majoration des indices de Sobol totaux par les coefficients DGSM. Toutefois, les DGSM ont ici l'avantage de garder un sens grâce à la forme mathématique du gradient du réseau. On remarque que si les poids d'entrée $\boldsymbol{w'}$ et de sortie $\boldsymbol{w''}$ associés à une entrée x_i sont proches de zéro, alors ceci signifie que le coefficient DGSM est également proche de zéro. C'est donc une bonne traduction mathématique d'une caractéristique structurelle d'un réseau de neurones.

En ce qui concerne l'application de cette analyse de sensibilité, elle est réalisée une fois qu'un premier réseau de neurones entier a été construit. Les coefficients DGSM peuvent ainsi être calculés sur le réseau de neurones non rebouclé. Ensuite, une nouvelle optimisation des poids et la sélection du nombre de neurones optimal peuvent être réexécutées, cette fois-ci avec moins d'entrées, et donc moins de poids à calculer.

9.3 Adaptation à la présence d'un nombre réduit de trajectoires d'apprentissage

9.3.1 Validation croisée et répartition des trajectoires en *folds*

Pour rappel, la validation croisée est une technique provenant de l'apprentissage statistique [Efron, 1983]. Lorsque la taille du DOE est réduite, cette méthode permet d'approcher l'erreur de généralisation (idée définie en section 3.1.2) tout en utilisant toutes les trajectoires disponibles au cours de l'apprentissage. L'idée est de décomposer le DOE en V groupes aussi appelés *folds* et de construire autant de modèles que de *folds*. À chaque calcul de poids sur les $V-1$ groupes, on teste le réseau ainsi construit sur le $V^{ème}$ non utilisé pour calculer les poids. Ainsi en moyennant les erreurs de test, on obtient l'erreur de validation croisée qui sert d'estimateur de l'erreur de généralisation. Ceci est détaillé à l'algorithme 3.2.

Une première modification introduite ici de la méthode classique de validation croisée est d'utiliser comme prédicteur la moyenne de la sortie des V modèles calculés sur chacun des *folds*.

$$\hat{f} = \frac{1}{V}\sum_{v=1}^{V} \hat{f}_v$$

Les études [Lozzo *et al.*, 2013] et [Lecué *et al.*, 2012] ont montré que l'utilisation en prédiction de cette agrégation des V modèles permettait de diminuer l'erreur commise par rapport à chacun des V modèles pris séparément.

À présent, intéressons nous à l'effet de la construction de sous-ensembles au DOE pour appliquer la validation croisée. Il est évident que la manière dont les points d'apprentissage sont répartis dans les *folds* influe de manière non négligeable sur la capacité qu'a l'erreur de validation croisée d'être une bonne estimation de l'erreur de généralisation. En particulier dans le cas extrême où tous les *folds* sont identiques, l'erreur de validation croisée se comporte comme une erreur d'apprentissage. Dans la section 3.1, nous avons montré que cette dernière ne permettait pas de trouver le nombre de neurones permettant le compromis entre le biais et la variance du modèle en prédiction.

La section 3.3.2.2 détaille deux répartitions des trajectoires existantes dans la littérature. La première repose sur la répartition des trajectoires entières. La seconde, appelée répartition par *sous-trajectoires*, consiste à ne plus considérer les trajectoires comme un tout insécable mais à répartir chaque pas de temps dans des *folds* différents, comme représenté sur la figure 3.19. Dans le cas où le DOE est composé d'un nombre réduit de trajectoires, la première répartition ne permet pas de garantir la robustesse puisqu'il peut y avoir une grande variabilité entre les *folds* et de ce fait, certains d'entre eux remplissent mal l'espace des entrées.

La seconde méthode n'a pas ce problème là puisque chacun des *folds* est construit à partir d'information provenant de toutes les trajectoires. Néanmoins, cette répartition par *sous-trajectoires* a pour désavantage de construire V groupes qui peuvent être très proches les uns des autres. Comme expliqué précédemment avec le cas extrême où ils sont tous identiques, cette proximité entre les échantillons de différents *folds* implique que l'erreur de validation croisée se comporte comme une erreur d'apprentissage, ce qui ne permet pas de choisir la complexité optimale.

La principale raison pour laquelle les *folds* peuvent être très proches est la régularité

du phénomène à prédire ainsi que la discrétisation temporelle fine utilisée. De ce fait, pour améliorer la technique de répartition par *sous-trajectoires*, il est nécessaire de filtrer les redondances des trajectoires d'apprentissage et de ne plus les utiliser, ni dans l'apprentissage ni pour tester les modèles construits.

Pour cela, il faut utiliser la variation temporelle des sorties par rapport au temps. Puisque, lors de l'apprentissage, nous travaillons sortie par sortie, nous pouvons utiliser la dérivée temporelle de la sortie étudiée par rapport au temps pour décider si un pas de temps d'une trajectoire doit être conservé ou non. La dérivée est calculée par différences finies en moyennant la valeur absolue de la dérivée à gauche et la dérivée à droite en chaque temps t^k sur les trajectoires d'apprentissage. Ceci signifie qu'une dérivée $\frac{dy_j^{k,l}}{dt} \simeq 0$ implique que les points $y_j^{k-1,l}$ à gauche et $y_j^{k+1,l}$ à droite ont une valeur très proche de celle au pas de temps étudié $y_j^{k,l}$. De ce fait, ce point peut être retiré du DOE. Plus précisément, on retire uniquement ce pas de temps du calcul de l'erreur lors du calcul des poids (détaillé en section 9.2). En effet, chaque pas de temps est inévitablement utilisé pour prédire les trajectoires, par construction du réseau de neurones récurrent.

Si l'on se limite à ce filtre, il peut arriver que l'on se retrouve avec de longs intervalles en temps sans points utilisés pour calculer l'erreur à l'apprentissage. Or, il se peut que ces points apportent une information sur la constance d'une sortie par rapport à la variation des entrées exogènes. De plus, puisque nous avons peu de points d'apprentissage à disposition, nous ne pouvons nous permettre de trop en négliger. De ce fait, nous récupérons des points filtrés dans le cas où de trop longs intervalles de temps consécutifs se retrouvent sans points. La durée maximum durant laquelle aucun point n'est utilisé pour l'erreur dépend du cas. Ici, nous l'exprimons en $k\Delta t$. Dans notre application, nous choisissons $k = 5$ de sorte à diviser au maximum la taille du DOE par 5.

Enfin, une fois ces points filtrés, les points restants sont répartis dans les folds séquentiellement comme précédemment. Ceci est illustré sur la figure 9.7.

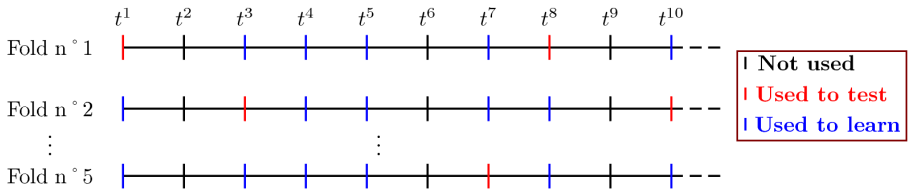


FIGURE 9.7 – Illustration de la répartition des points dans les folds après filtre des redondances grâce à la dérivée temporelle de la sortie par rapport au temps.

9.3.2 Piste pour les plans d'expériences spatio-temporels

En ce qui concerne la génération du plan d'expériences, la méthode choisie consiste à faire du *space-filling*, c'est-à-dire à remplir au mieux l'espace des entrées. Toutefois, dans le cas de réseau de neurones rebouclé, certaines des entrées correspondent à des sorties rebouclées. Nous n'avons donc aucun contrôle et aucun *a priori* sur les entrées de $\hat{f}$ correspondantes. De ce fait, les techniques classiques appliquées à la construction de DOE en statique ne sont pas applicables ici.

Le choix qui est fait ici est de générer les entrées exogènes $\boldsymbol{u}$ aléatoirement sur une ou des bases de fonctions tels que des sinusoïdes ou des créneaux. L'utilisation d'une base de

fonction permet de se ramener à la génération de paramètres ne dépendant plus du temps, et donc pour lesquels nous pouvons utiliser des techniques du type LHS (détaillée dans la section 3.1.3). Par exemple, afin de remplir l'espace des entrées au mieux, nous pouvons utiliser des sinusoïdes. En effet, elles présentent l'avantage de bien remplir l'espace des entrées du réseau non rebouclé $\hat{f}$ et donc de fournir des trajectoires d'apprentissage plus informatives. Afin de générer aléatoirement des entrées exogènes sur la base de fonctions sinusoïdales, nous les paramétrons de la manière suivante :

$$u(t) = C_1 \sin\left(C_2\left(t + t_0\right)\right) + C_3 \tag{9.8}$$

Les paramètres C_1, C_2 et C_3 sont tirés aléatoirement par LHS. Les bornes de l'espace dans lequel ils sont générés dépendent quant à elles du cas. Par exemple, pour le débit d'extraction de l'air, ces bornes sont choisies pour prendre en compte le cas de panne dans les trajectoires d'apprentissage. Pour cela, le domaine de variation des paramètres d'amplitude de la sinusoïde (C_1 et C_3) est fixé de sorte à contenir le domaine de variation du débit des trajectoires à prédire.

Dans le cas de trajectoires de type créneau, ce qui correspond davantage à la forme des entrées exogènes d'un plan de vol comme illustré sur la figure 9.1, la paramétrisation est réalisée comme suit :

$$u(t) = \begin{cases} C_2 & \text{si } \exists k \in \mathbb{N} \ : \ 2kC_1 \leqslant t < (2k+1)C_1 \\ C_3 & \text{si } \exists k \in \mathbb{N} \ : \ (2k-1)C_1 \leqslant t < 2kC_1 \end{cases} \tag{9.9}$$

Afin de construire un plan d'expériences de type *space filling*, Derek Dinart [Dinart, 2014], dans son mémoire de recherche de M1, a mis en place des techniques permettant de choisir les trajectoires les plus informatives parmi un jeu de trajectoires déjà évaluées sur le modèle de référence. Son idée, proche de celle du critère d'enrichissement Maximin [Franco, 2008], consiste à partir de la trajectoire la plus *étalée* et à choisir ensuite séquentiellement celles qui remplissent au mieux l'espace des entrées de $\hat{f}$, d'où le besoin d'évaluer les entrées exogènes sur le modèle de référence afin de connaître la valeur des sorties rebouclées. Toutefois, une trajectoire est un ensemble de points. De ce fait, pour comparer le DOE courant et l'échantillon que l'on souhaite rajouter, la distance entre deux ensembles de points doit être calculée. L'idée de Dinart est d'utiliser la distance min, la distance max et *l'étalement* de la trajectoire à rajouter pour choisir la plus informative. Nous ne détaillons pas davantage ces techniques puisqu'elles nécessitent de générer, et de les estimer sur le modèle de référence, un certain nombre de trajectoires avant de choisir les meilleures. Ceci n'est pas accessible ici.

9.4 Application au cas d'un équipement électronique

Dans cette section, nous mettons en place la technique de construction présentée à la section 9.1.3 améliorée par les techniques présentées aux sections 9.2 pour l'optimisation des poids et 9.3 pour la nouvelle répartition des trajectoires dans les *folds* pour la validation croisée. Cette méthodologie est appliquée au cas test présenté à la section 9.1.1.2. Pour rappel, ce cas test a $N_u = 4$ entrées exogènes et $N_y = 6$ sorties rebouclées.

Afin de mettre en place la construction détaillée à l'algorithme 9.2, nous devons d'abord construire un DOE. La génération des trajectoires est expliquée à la section 9.3.2. 37 trajectoires de 300 s sont générées aléatoirement à partir de créneaux et de sinusoïdes. Le

pas de temps de la simulation transitoire utilisé ici est de $\Delta t = 1$ s. Un exemple de sorties obtenues avec des créneaux est donné sur la figure 9.8 et un exemple de sorties obtenues avec des sinusoïdes est donné en figure 9.9. Ces deux figures permettent en particulier de remarquer que les sorties sont non-linéaires et ne sont pas de la même forme que les entrées exogènes.

En ce qui concerne les trajectoires de test, en plus du profil de vol de la figure 9.1 (d'une durée de 42740 s), nous avons également 36 trajectoires de test de 300 s également générées aléatoirement par un mélange de sinusoïdes et de créneaux.

Avec le DOE de 37 trajectoires de 300 s, nous pouvons à présent lancer la construction de l'algorithme 9.2, en initialisant d'abord les poids par l'algorithme 9.1. C'est également à cette étape que nous choisissons le nombre de neurones optimal. Pour illustrer le fait que la modification de la répartition des trajectoires est effective, nous comparons le comportement de l'erreur de validation croisée avec la technique de *sous-trajectoires* proposée par [Lozzo *et al.*, 2013] et notre modification introduite à la section 9.3. Pour montrer que le nombre de neurones choisi correspond au nombre minimisant l'erreur de généralisation, nous comparons ces deux erreurs de validation croisée à l'erreur sur notre base de test (de 36 trajectoires et le plan de vol). Le résultat est montré sur la figure 9.10. Elle met en évidence ce qui était expliqué dans la section 9.3 : l'erreur de validation croisée n'arrête pas de décroître si les *folds* sont trop proches, ce qui est le cas ici avec la technique des *sous-trajectoires*. En retirant des points redondants de l'apprentissage, on remarque cette fois-ci que le nombre de neurones qui minimise l'erreur de validation croisée ($C = 9$) correspond à celui qui minimise l'erreur de généralisation. Ceci montre que l'erreur de validation croisée avec la répartition des trajectoires par *sous-trajectoires* modifiées permet d'avoir une bonne approximation de l'erreur de généralisation.

À présent, regardons l'effet des itérations de l'algorithme 9.2 sur l'erreur de généralisation pour mettre en évidence l'intérêt de l'optimisation multi-niveaux. Pour cela, comparons le résultat d'un réseau construit par optimisation multi-niveaux avec un réseau construit par une optimisation classique sur les 37 trajectoires de test (36 aléatoires de 300 s et le plan de vol). Le graphique de la figure 9.11 compare l'évolution de l'erreur relative entre ces deux méthodes. On remarque d'abord que trois itérations suffisent pour que le critère d'arrêt soit atteint. Ensuite, cette figure montre que les itérations améliorent le résultat du réseau de neurones en prédiction. En particulier, au bout des trois itérations, on voit que sur 90% des points de la base de test, l'erreur commise par le réseau est inférieure à 1% d'erreur relative et que l'erreur maximale est de 3%, ce qui est dans le seuil d'erreur relative acceptable pour l'industriel qui est de 5%. En ce qui concerne l'optimisation classique des poids, on remarque qu'elle est totalement inefficace dans ce cas. Elle ne permet pas de retourner un réseau de neurones donnant un résultat satisfaisant : en effet, plus de 30% de l'erreur relative en prédiction est supérieure à 10%, ce qui est inacceptable.

Pour illustrer la précision obtenue sur le profil de vol par un réseau construit par optimisation multi-niveaux des poids, la figure 9.12 compare les sorties prédites par le réseau et les sorties mesurées, c'est-à-dire calculées sur le modèle de référence. Les deux courbes se superposent pratiquement. En particulier, on peut voir que le passage de la convection forcée à naturelle puis de la naturelle à la forcée autour du temps $t = 33000$ s est bien gérée par le réseau de neurones qui y commet peu d'erreurs. Ceci met en évidence le fait que les réseaux de neurones récurrents sont capables de prédire des phénomènes fortement non-linéaires.

Enfin, sur le réseau précédemment construit, nous pouvons calculer les coefficients

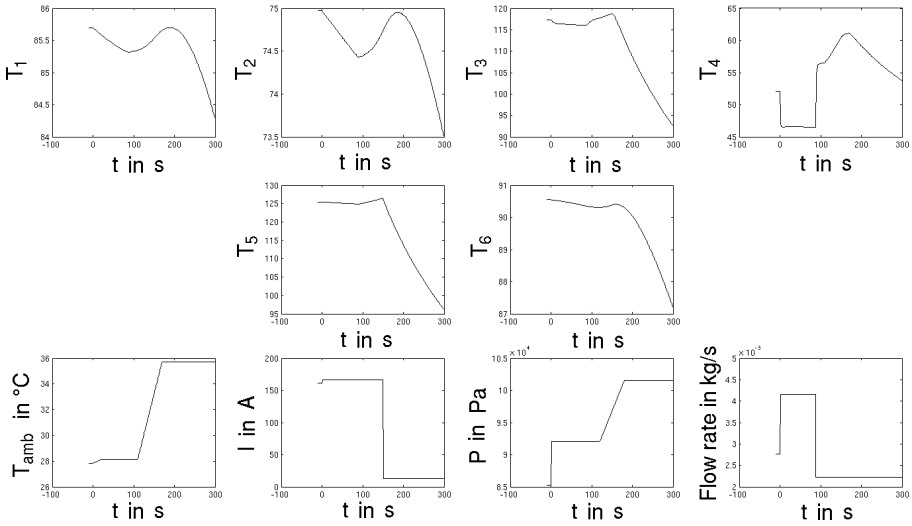


FIGURE 9.8 – Illustration des entrées exogènes générées aléatoirement par créneaux (représentées en bas) avec les 6 températures d'intérêt.

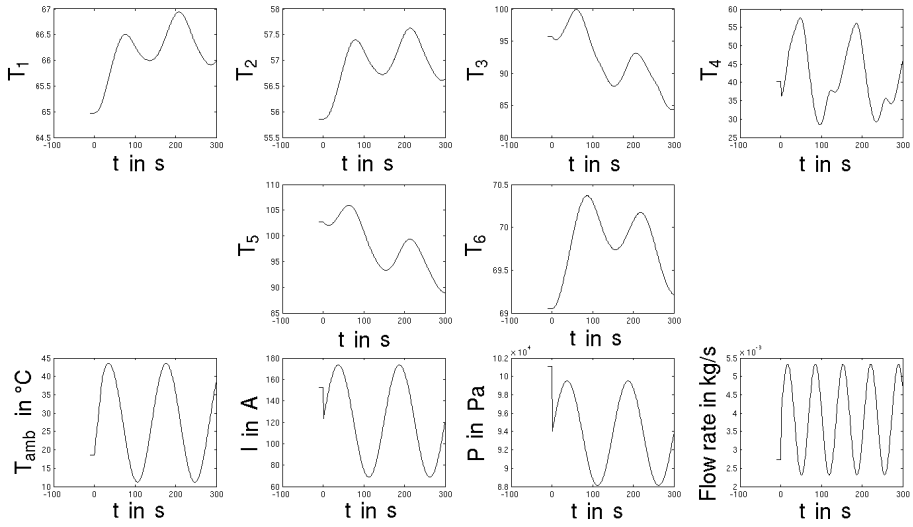


FIGURE 9.9 – Illustration des entrées exogènes générées aléatoirement par sinusoïde (représentées en bas) avec les 6 températures d'intérêt.

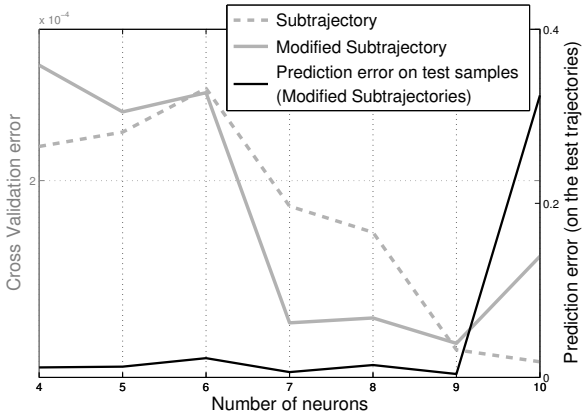


FIGURE 9.10 – Illustration sur le cas test de l'évolution de l'erreur de validation croisée avec le nombre de neurones avant et après la modification sur la répartition des trajectoires. Comparaison avec l'erreur de généralisation estimée sur la base de test représentée en noir.

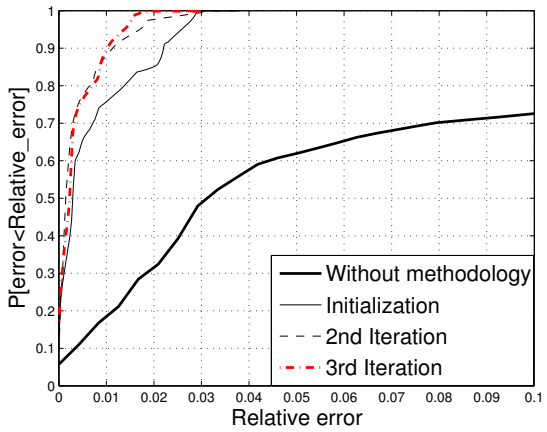


FIGURE 9.11 – Illustration sur le cas test de la probabilité que l'erreur relative commise par toutes les sorties soit plus grande que l'erreur relative en abscisse. Cette probabilité est calculée sur les 37 trajectoires de test dont le plan de vol.

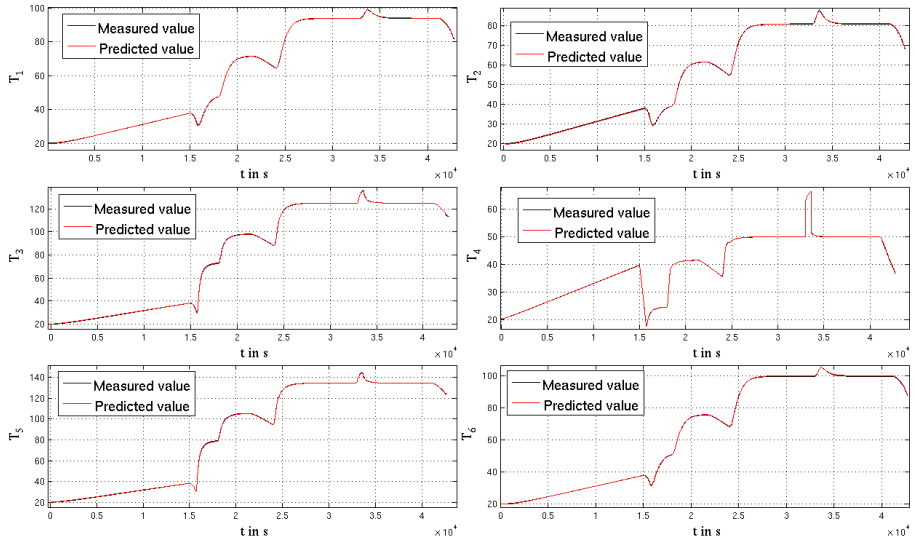


FIGURE 9.12 – Résultat obtenu par un réseau construit par optimisation multi-niveaux sur les entrées exogènes données à la figure 9.1. Comparaison avec les sorties obtenues sur le modèle de référence.

DGSM par Monte-Carlo sur le gradient de la sortie du réseau de neurones non bouclé (donné à l'équation (9.7)). La figure 9.13 illustre la valeur de ces coefficients uniquement pour les entrées correspondant à des sorties rebouclées, les coefficients DGSM ν_i étant estimés par la formule de l'équation (4.35). De cette figure, on en déduit à l'équation (9.10) la matrice où 1 signifie que l'entrée (en ordonnée) est utilisée pour prédire la sortie (en abscisse) et 0 que l'entrée peut être retirée pour prédire la sortie en abscisse.

$$\begin{array}{r} T_{amb} \rightarrow \\ P \rightarrow \\ Q \rightarrow \\ I \rightarrow \\ \hat{T}_1 \rightarrow \\ \hat{T}_2 \rightarrow \\ \hat{T}_3 \rightarrow \\ \hat{T}_4 \rightarrow \\ \hat{T}_5 \rightarrow \\ \hat{T}_6 \rightarrow \end{array} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ \hat{T}_1 & \hat{T}_2 & \hat{T}_3 & \hat{T}_4 & \hat{T}_5 & \hat{T}_6 \end{pmatrix} \tag{9.10}$$

On remarque que la sortie correspondant à l'air (T_4) n'est pas utile pour prédire les autres sorties. D'un point de vue physique, ceci signifie que l'on peut en effet se contenter du débit d'extraction de l'air et de la température ambiante dans la cavité pour quantifier les échanges thermiques entre l'air à l'intérieur de l'équipement et la structure. De plus, si l'on s'affranchit de la température T_4, on peut également retirer la température T_2 qui correspond à la température de la paroi opposée au composant critique. En effet on voit à l'équation (9.10) qu'elle n'était nécessaire que pour prédire la température T_4. Ainsi, si la connaissance de cette température n'est pas nécessaire pour l'utilisateur, elle peut également être retirée de la construction. C'est le cas ici puisque nous ne sommes intéressés que par la température du composant critique. On peut donc retirer les températures T_4

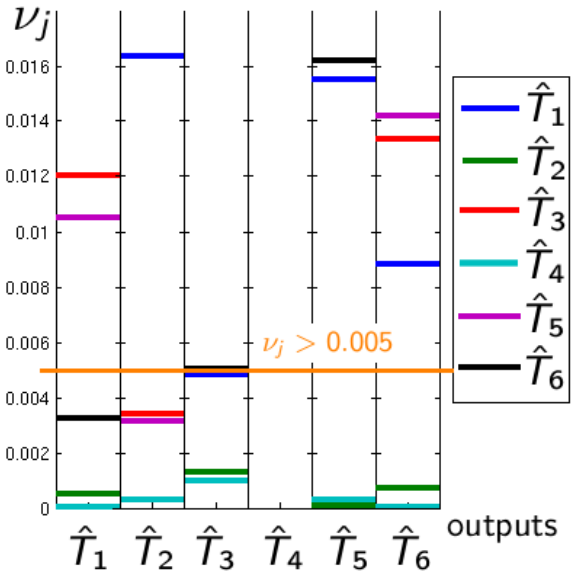


FIGURE 9.13 – Valeur des coefficients DGSM normalisés à 1 pour chacune des entrées correspondant à des sorties rebouclées du réseau de neurones. En abscisse, chacune des sorties sont données. En orange le seuil en dessous duquel on retire l'entrée correspondante.

et T_2 du processus et reconstruire un réseau de neurones.

Après reconstruction, on remarque que le nombre de poids à optimiser est nettement inférieur puisque l'on passe de 695 poids à 409. Par ailleurs, le nombre de neurones nécessaire est également inférieur, passant de 53 à 35. La figure 9.14 compare le résultat avant et après le retrait des sorties. On remarque que l'erreur est globalement inférieure pour le réseau de neurones construit après analyse de sensibilité, ceci met en évidence deux choses :

1. Les sorties retirées ne sont effectivement pas influentes, ce qui montre que les coefficients DGSM ont du sens, malgré la corrélation de certaines des entrées. Ceci est en particulier dû au fait que ces coefficients traduisent une propriété structurelle des réseaux comme expliqué en fin de section 9.2.2. Ils peuvent donc être utilisés à condition que le premier réseau de neurones construit soit suffisamment précis. Dans le cas contraire, puisque les DGSM quantifient les effets sur le réseau de neurones, si ce dernier n'est pas bon, les conclusions que l'on tire des coefficients DGSM ne sont pas valables pour le modèle de référence.
2. En diminuant le nombre d'entrée, on remarque que l'on obtient un meilleur résultat alors que l'on fournit moins d'information au réseau. C'est une preuve supplémentaire que la dimension d'entrée pénalise effectivement le calcul des poids optimaux.

9.5 Conclusions sur la construction de modèles de substitution spatio-temporels par réseaux de neurones récurrents

Pour résumer, nous avons proposé une méthode de construction de modèles de substitution spatio-temporels par réseaux de neurones récurrents permettant de répondre au

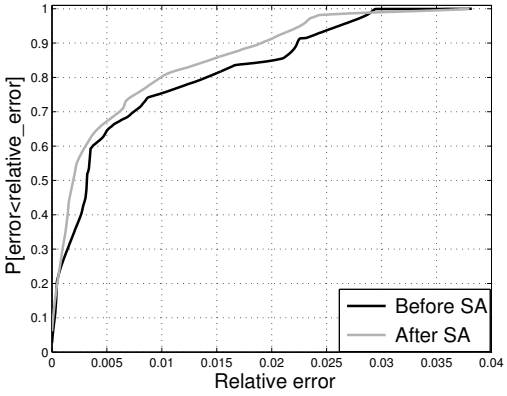


FIGURE 9.14 – Illustration sur le cas test de la probabilité que l'erreur relative commise par toutes les sorties soit plus grande que l'erreur relative en abscisse. Cette probabilité est calculée sur les 37 trajectoires de test dont le plan de vol. Comparaison entre le réseau construit avec toutes les entrées et celui construit après analyse de sensibilité et donc avec moins d'entrées.

cadre imposé par le cas industriel et donné en section 9.1.2. Pour cela, nous sommes partis de la construction proposée par [Lozzo *et al.*, 2013], nous en avons identifié les limites par rapport au cadre imposé en section 9.1.3. La première est la dimension importante du problème d'optimisation à résoudre pour construire le réseau de neurones. La seconde est le peu de trajectoires d'apprentissage disponibles.

Pour répondre à cela, l'optimisation des poids est décomposée par une technique d'optimisation multi-niveaux à la section 9.2. Puis, la validation croisée et la répartition des trajectoires qu'elle nécessite est adaptée au problème en section 9.3. L'application de ces techniques ont montré sur le cas industriel en section 9.4 que :

— La diminution de la dimension du problème d'optimisation permettait effectivement d'améliorer la qualité de la solution retournée. En particulier, l'analyse de sensibilité appliquée sur le réseau de neurones non rebouclé comme expliqué en section 9.2.2, a permis de mettre en évidence l'importance de la réduction de dimension de la minimisation de l'erreur commise par le réseau.

— La validation croisée est sensible à la manière dont le DOE est scindé. En particulier, si les *folds* sont trop proches, ou ne remplissent pas suffisamment l'espace des entrées, l'erreur de validation croisée ne fournit pas une bonne approximation de l'erreur de généralisation.

Toutefois, la construction proposée présente des limites. Voici celles que nous avons identifiées.

— La principale limite concerne l'absence de critère d'enrichissement *a priori* du DOE. En effet, certaines des entrées de $\hat{f}$ sont des sorties rebouclées. Or, l'utilisateur n'a de contrôle que sur les entrées exogènes et ne peut de ce fait pas placer un point dans l'espace des entrées de $\hat{f}$ à l'endroit demandé par les méthodes de DOE présentée en section 3.1.3. Une piste envisageable serait de partir d'un DOE initial réduit (comme à l'algorithme 3.5), de construire un premier modèle à partir de ce dernier, et d'utiliser les prédictions du modèle pour générer de nouvelles trajectoires et placer celles qui sont le plus distantes du DOE actuel en utilisant les critères mis

en place par Dinart dans son étude [Dinart, 2014].

— Une autre limite concerne la dépendance de la méthode proposée à la physique étudiée. En effet, cette dernière permet de choisir l'ordre en temps utilisé. Dans le cas où le phénomène à prédire est d'un ordre en temps supérieur, l'optimisation multi-niveaux des poids et le filtre des points pour la division du DOE en 5 *folds* restent encore valable. Cependant, ceci augmente davantage la dimension des poids à calculer, complexifiant ainsi la construction.
Cette dépendance à la physique pose également problème pour choisir les points d'intérêt que nous souhaitons prédire. Pour cela, l'expertise d'un ingénieur reste indispensable et il est difficile d'envisager des techniques entièrement automatisables.

— La dernière limite est la conséquence de la deuxième limite : dans le cas de physiques nécessitant des ordres en temps supérieurs, le problème d'optimisation est de dimension plus grande encore. Il peut donc être profitable d'investiguer d'autres méthodes d'optimisation des poids plutôt que les méthodes basées sur le gradient. En particulier, les méthodes basées sur les filtres de Kalman étendu [Julier et Uhlmann, 1997] peuvent être une alternative intéressante. Leur application à l'apprentissage de réseaux de neurones est détaillée dans les études [Trebatický et Pospíchal, 2008], [Wang et Huang, 2011].

Chapitre 10

Développements théoriques sur les estimateurs du quantile et du superquantile par échantillonnage préférentiel dans le cadre donné par l'optimisation

Sommaire

L'objectif de cette partie est d'étudier le comportement asymptotique théorique des estimateurs par échantillonnage préférentiel des mesures de risque que sont le quantile et le superquantile. Les démonstrations de convergence de ces estimateurs ont déjà été données (au théorème 4.1 pour le quantile et au théorème 4.2 pour le superquantile). Nous étudions ici l'évolution du comportement de ces estimateurs en différents jeux de paramètre de contrôle (cadre imposé par l'optimisation). La section 10.2 a donc pour objet d'étudier la corrélation de ces erreurs. Par la suite, la section 10.3 s'intéresse à la convergence de l'estimateur fonctionnel vers un processus aléatoire lorsque la taille de l'échantillon utilisé tend vers l'infini.

Dans la section 10.4, nous vérifions que l'estimateur par échantillonnage préférentiel de nos mesures de risque converge encore dans le cas de données tirées selon différentes lois biaisées. En effet, la problématique de réutilisation des appels à f pour l'estimation

de la mesure de robustesse en deux points $\boldsymbol{x}$ et $\boldsymbol{x}'$ distincts a été développée au chapitre 8.

Enfin, l'estimateur par échantillonnage préférentiel du quantile peut ne pas exister puisqu'il consiste à comparer une somme non normalisée à 1 et une valeur $\alpha \in]0,1[$. Il faut donc s'assurer de son existence à partir d'une certaine taille d'échantillon $N^0 \in \mathbb{N}$.

10.1 Preuve de l'existence de l'estimateur du quantile par échantillonnage préférentiel

Pour estimer le quantile par échantillonnage préférentiel, on utilise l'estimateur donné à l'équation (10.1).

$$\tilde{H}_{N_{IS}}(y;\ \boldsymbol{x}^k) = \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}^k)\ \mathbf{1}_{f(\boldsymbol{x}^k,\tilde{\boldsymbol{\xi}}^i)\leqslant y} \tag{10.1}$$

et $\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}^k) = \inf\left\{y\ :\ \tilde{H}_{N_{IS}}(y;\ \boldsymbol{x}^k) \geqslant \alpha\right\}$.

Dans cette partie, nous prouvons l'existence de cet estimateur. Pour cela, il faut que $\exists y \in \mathbb{R}\ :\ \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}^k)\ \mathbf{1}_{f(\boldsymbol{x}^k,\tilde{\boldsymbol{\xi}}^i)\leqslant y} \geqslant \alpha$ dans le cas où $0 < \alpha < 1$. Il est important de le vérifier puisque cette somme n'est pas « normalisée » à 1. Cette démonstration se fait par étape, en passant par deux lemmes.

Lemme 10.1. *Si* $\forall \boldsymbol{x} \in \mathcal{D}_x$, $\mathbb{E}[(L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x})^2] < \infty$, *alors :*

$\forall \boldsymbol{x} \in \mathcal{D}_x$, $\forall \alpha \in]0,1[$, $\forall \varepsilon > 0$, $\exists N^0 \in \mathbb{N}^\star$, *tel que* $\forall N_{IS} \geqslant N^0$ *:*

$$\mathbb{P}\left(\frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \leqslant \alpha\right) \leqslant \varepsilon$$

Preuve.
Ceci se montre par l'application de la loi des grands nombres sur la suite de variables indépendantes $\left(L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x})\right)_{i\in\mathbb{N}^\star}$. En effet, $\forall \boldsymbol{x} \in \mathcal{D}_x$ ces variables aléatoires sont de variance finie par hypothèse. De plus, elles sont de même espérance $\mathbb{E}\big(L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x})\big) = 1$, par construction des rapports de vraisemblance. On a donc par la loi faible des grands nombres [Saporta, 2006, p.277] :

$$\forall \varepsilon' > 0\ ,\ \lim_{N_{IS}\to\infty} \mathbb{P}\left[\left|\frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) - 1\right| > \varepsilon'\right] = 0$$

Posons $\mathcal{S} = \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x})$. L'assertion précédente équivaut à :

$\forall \varepsilon' > 0$, $\forall \varepsilon > 0$, $\exists N^0 \in \mathbb{N}^\star$ tel que $\forall N_{IS} > N^0$

$$\mathbb{P}\left[|\mathcal{S} - 1| > \varepsilon'\right] \leqslant \varepsilon \tag{10.2}$$

Définissons les deux évènements A et B de la manière suivante :

$$\begin{cases} A &:= \{|\mathcal{S} - 1| > \varepsilon'\} \\ B &:= \{1 - \mathcal{S} > \varepsilon'\} \end{cases}$$

On a $|\mathcal{S}-1| \geqslant 1-\mathcal{S}$, ce qui implique que $B \subset A$ et donc $\mathbb{P}(B) \leqslant \mathbb{P}(A)$. L'équation (10.2) implique donc :

$$\mathbb{P}\left[1-\mathcal{S} > \varepsilon'\right] \leqslant \varepsilon \Rightarrow \mathbb{P}\left[\mathcal{S}-1 < -\varepsilon'\right] \leqslant \varepsilon \Rightarrow \mathbb{P}\left[\mathcal{S} < 1-\varepsilon'\right] \leqslant \varepsilon$$

En résumé nous avons donc :
$\forall \varepsilon' > 0$, $\forall \varepsilon > 0$, $\exists N^0 \in \mathbb{N}^\star$ tel que $\forall N_{IS} > N^0$

$$\mathbb{P}\left[\mathcal{S} \leqslant 1-\varepsilon'\right] \leqslant \varepsilon$$

En prenant $\varepsilon' = 1-\alpha$, ce qui est possible car $\alpha \in]0,1[$, nous obtenons le résultat annoncé. □

Nous pouvons à présent montrer que pour tout $\boldsymbol{x} \in \mathcal{D}_x$, que la probabilité que le quantile existe lorsque le nombre d'échantillons tend vers l'infini est égale à 1. Pour cela, définissons l'évènement $\mathcal{A}^{N_{IS}}$ de la manière suivante :

$$\mathcal{A}^{N_{IS}} = \left\{\tilde{q}_\alpha^{N_{IS}} \text{ existe}\right\}$$

Lemme 10.2.

$$\mathcal{A}^{N_{IS}} \iff \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \alpha$$

Preuve.

(⇐) Supposons $\frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \alpha$. Pour prouver que $\tilde{q}_\alpha^{N_{IS}}$ existe, il suffit donc de montrer qu'il existe un $y \in \mathbb{R}$ tel que $\mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\leqslant y} = 1$. $\forall N_{IS} \in \mathbb{N}$ en prenant $y = \sup_{i=1,\cdots,N_{IS}} f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i) + 1$, on obtient le résultat annoncé.

(⇒) Supposons que $\tilde{q}_\alpha^{N_{IS}}$ existe, alors ceci signifie qu'il existe un $y \in \mathbb{R}$ tel que

$$\frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\leqslant y} \geqslant \alpha$$

Or, $\mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\leqslant y} \leqslant 1$. On a donc :

$$\frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\leqslant y}$$

On en déduit :

$$\frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \alpha$$

□

Théorème 10.1 (Théorème d'existence de l'estimateur du quantile par IS)**.**
Si $\forall \boldsymbol{x} \in \mathcal{D}_x$, $\mathbb{E}[(L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x})^2] < \infty$, *alors :*

$$\mathbb{P}\left(\mathcal{A}^{N_{IS}}\right) \underset{N_{IS}\to\infty}{\to} 1$$

Preuve.
Du lemme 10.2, on peut en déduire que

$$\mathbb{P}\left(\mathcal{A}^{N_{IS}}\right) = \mathbb{P}\left(\frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \alpha\right)$$

Or,

$$\mathbb{P}\left(\frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \alpha\right) = 1 - \mathbb{P}\left(\frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) < \alpha\right)$$

Or le lemme 10.1 implique que

$$\mathbb{P}\left(\frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) < \alpha\right) \underset{N_{IS}\to\infty}{\to} 0$$

On peut donc en déduire que :

$$\mathbb{P}\left(\frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x}) \geqslant \alpha\right) \underset{N_{IS}\to\infty}{\to} 1$$

Et donc que $\mathbb{P}\left(\mathcal{A}^{N_{IS}}\right) \underset{N_{IS}\to\infty}{\to} 1$. □

10.2 Étude de la corrélation des erreurs en d points d'optimisation

Dans le cadre particulier de l'optimisation où le quantile et le superquantile doivent être estimés en différents points $\boldsymbol{x} \in \mathcal{D}_x$, nous n'avons pas trouvé dans la littérature de démonstrations de convergence des estimateurs par échantillonnage préférentiel. Celles que nous avons trouvées sont données aux théorèmes 4.1 et 4.2. Toutefois, elles démontrent uniquement la convergence de ces estimateurs autour de chaque point $\boldsymbol{x} \in \mathcal{D}_x$.

L'objectif de cette partie est donc d'étudier le comportement asymptotique d'une mesure de risque évaluée simultanément en d différents points $\left(\boldsymbol{x}^k\right)_{1\leqslant k\leqslant d}$. Il est notamment possible d'estimer la corrélation de l'erreur commise entre ces d différents points. Dans cette partie, on se place dans le cas où la vraie densité des incertitudes en entrée peut dépendre du point $\boldsymbol{x}$ où l'effet des incertitudes est calculé. Ce qui signifie que le rapport de vraisemblance $L(\tilde{\boldsymbol{\xi}}^i;\ \boldsymbol{x})$ dépend du point $\boldsymbol{x}$. En revanche, dans tout ce chapitre, la loi biaisée permettant de générer les incertitudes $\left(\tilde{\boldsymbol{\xi}}^1, \cdots, \tilde{\boldsymbol{\xi}}^{N_{IS}}\right)$ est la même quel que soit le point où l'on se trouve.

Cas du quantile

Théorème 10.2. *Si* $\forall k \in \{1..d\}$ *on a* $\mathbb{E}[(L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^k)^{3+\delta}] < \infty$ *avec* $\delta > 0$ *et* $H\left(\bullet;\ \boldsymbol{x}^k\right)$ *est différentiable en* $q_\alpha(\boldsymbol{x}^k) = H^{-1}\left(\alpha;\ \boldsymbol{x}^k\right)$ *avec* $H'\left(q_\alpha(\boldsymbol{x}^k);\ \boldsymbol{x}^k\right) = h_Y(q_\alpha(\boldsymbol{x}^k);\ \boldsymbol{x}^k) > 0$. *Enfin, si les d jeux de paramètres de contrôle sont distincts deux à deux, alors* $\forall d \geqslant 2$ *:*

$$\sqrt{N_{IS}} \begin{pmatrix} \tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}^1) - q_\alpha(\boldsymbol{x}^1) \\ \cdots \\ \tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}^d) - q_\alpha(\boldsymbol{x}^d) \end{pmatrix} \xrightarrow[N_{IS}\to\infty]{\mathcal{L}} \mathcal{N}(0, DSD)$$

Avec $(S_{ij})_{i=1..d,j=1..d} = \left(\tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^i)L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^j)\mathbf{1}_{f(\boldsymbol{x}^i,\tilde{\boldsymbol{\xi}}^1)\leqslant q_\alpha(\boldsymbol{x}^i)}\mathbf{1}_{f(\boldsymbol{x}^j,\tilde{\boldsymbol{\xi}}^1)\leqslant q_\alpha(\boldsymbol{x}^j)}\right] - \alpha^2\right)_{i=1..d,j=1..d}$

et $D = \begin{pmatrix} \frac{1}{h_Y(q_\alpha(\boldsymbol{x}^1);\ \boldsymbol{x}^1)} & 0 & \cdots & 0 \\ 0 & \frac{1}{h_Y(q_\alpha(\boldsymbol{x}^2);\ \boldsymbol{x}^2)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{1}{h_Y(q_\alpha(\boldsymbol{x}^d);\ \boldsymbol{x}^d)} \end{pmatrix}$

Preuve.
$\forall t = (t_1, \cdots, t_d) \in \mathbb{R}^d$, on a :

$$\tilde{\mathbb{P}}\left[\bigcap_{k=1}^{d}\left\{\sqrt{N_{IS}}(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}^k) - q_\alpha(\boldsymbol{x}^k)) \leqslant t_k\right\}\right] = \tilde{\mathbb{P}}\left[\bigcap_{k=1}^{d}\left\{\alpha \leqslant \tilde{H}_{N_{IS}}(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k)\right\}\right] \tag{10.3}$$

En introduisant les notations suivantes :

$$\begin{cases} \psi_{N_{IS}}^k(t) = \sqrt{N_{IS}}\left[\tilde{H}_{N_{IS}}(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k) - H\left(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k\right)\right] \\ a_{N_{IS}}^k(t) = \sqrt{N_{IS}}\left[\alpha - H\left(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k\right)\right] \end{cases}$$

Posons également :

$$\begin{aligned}(S_{N_{IS}})_{i=1..d,j=1..d} &= \tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^i)L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^j)\mathbf{1}_{f(\boldsymbol{x}^i,\tilde{\boldsymbol{\xi}}^1)\leqslant q_\alpha(\boldsymbol{x}^i)+t^i/\sqrt{N_{IS}}}\mathbf{1}_{f(\boldsymbol{x}^j,\tilde{\boldsymbol{\xi}}^1)\leqslant q_\alpha(\boldsymbol{x}^j)+t^j/\sqrt{N_{IS}}}\right]\\ &\quad -H\left(q_\alpha(\boldsymbol{x}^i)+\frac{t^i}{\sqrt{N_{IS}}};\ \boldsymbol{x}^i\right)H\left(q_\alpha(\boldsymbol{x}^j)+\frac{t^j}{\sqrt{N_{IS}}};\ \boldsymbol{x}^j\right)\end{aligned}$$

En reprenant ces notations pour l'équation (10.3), et en introduisant $\boldsymbol{\phi}^{N_{IS}} = \left(\phi_1^{N_{IS}},\cdots,\phi_d^{N_{IS}}\right) \sim \mathcal{N}(0, S_{N_{IS}})$, on obtient :

$$\begin{aligned}\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\alpha\leqslant \tilde{H}_{N_{IS}}(q_\alpha(\boldsymbol{x}^k)+\frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k)\right\}\right] &= \tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{a^k_{N_{IS}}(t)\leqslant \psi^k_{N_{IS}}(t)\right\}\right]\\ &\quad \underbrace{-\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\phi_k^{N_{IS}}\geqslant a^k_{N_{IS}}(t)\right\}\right]+\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\phi_k^{N_{IS}}\geqslant a^k_{N_{IS}}(t)\right\}\right]}_{=0!}\end{aligned}$$

Le lemme suivant permet de simplifier cette expression lorsque N_{IS} tend vers l'infini.

Lemme 10.3.
$\forall \boldsymbol{v}\in\mathbb{R}^d$, on a :

$$\left|\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\psi^k_{N_{IS}}(t)\leqslant v^k\right\}\right]-\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\phi_k^{N_{IS}}\leqslant v^k\right\}\right]\right| \underset{N_{IS}\to\infty}{\to} 0$$

Plus précisément, on a même :

$$\left|\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\psi^k_{N_{IS}}(t)\leqslant v^k\right\}\right]-\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\phi_k^{N_{IS}}\leqslant v^k\right\}\right]\right| = O\left(\frac{1}{\sqrt{N_{IS}}}\right)$$

Preuve.
D'après le théorème de l'étude de Bhattacharya [Bhattacharya, 1970, p.85], on a $\forall \boldsymbol{v}\in\mathbb{R}^d$:

$$\left|\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\psi^k_{N_{IS}}(t)\leqslant v_k\right\}\right]-\tilde{\mathbb{P}}\left[\bigcap_{k=1}^d\left\{\phi_k^{N_{IS}}\leqslant v_k\right\}\right]\right| \leqslant \frac{C_1(d,N_{IS})}{\sqrt{N_{IS}}}+C_2(d)\sup_{u\in\mathbb{R}^d}\left\{\int\omega_{I_u}\left(\mathcal{B}\left(\bullet,\frac{C_3(d,N_{IS})}{\sqrt{N_{IS}}}\right)\right)d\Phi\right\} \tag{10.4}$$

C_2 ne dépend que du nombre de point de design d qui est fixé. C_1 et C_3 dépendent également de d mais aussi de N_{IS} à cause de la dépendance en N_{IS} de $S_{N_{IS}}$.

Pour simplifier les écritures, posons :

$$\begin{cases} A^{N_{IS}}(d,N_{IS}) &= \dfrac{C_1(d,N_{IS})}{\sqrt{N_{IS}}}\\ B^{N_{IS}}(d,N_{IS}) &= \sup\limits_{u\in\mathbb{R}^d}\left\{\int\omega_{I_u}\left(\mathcal{B}\left(\bullet,\dfrac{C_3(d,N_{IS})}{\sqrt{N_{IS}}}\right)\right)d\Phi\right\}\end{cases}$$

Concernant la dépendance en N_{IS} de C_1 et C_3, elle est de la forme suivante :

$$\begin{cases} C_1(d,N_{IS}) &= C_1'(d)\left\|T_{N_{IS}}\right\|^{6+4\delta}\\ C_3(d,N_{IS}) &= C_3'(d)\left\|T_{N_{IS}}^{-1}\right\|\times\left\|T_{N_{IS}}\right\|^3\end{cases}$$

Concernant $T_{N_{IS}}$, cette matrice est définie de la manière suivante :

$$T_{N_{IS}}^T T_{N_{IS}} = S_{N_{IS}}$$

$T_{N_{IS}}$ existe puisque $S_{N_{IS}}$ est une matrice symétrique définie positive. De plus, d'après [Bhattacharya, 1970, p.85], on a :

$$\left\{ \begin{array}{rcl} \|T_{N_{IS}}\| & \leqslant & \left\{\mathrm{Tr}\left(S_{N_{IS}}\right)\right\}^{1/2} \\ \left\|T_{N_{IS}}^{-1}\right\| & \leqslant & \left\{\mathrm{Tr}\left(S_{N_{IS}}^{-1}\right)\right\}^{1/2} \end{array} \right.$$

Or, $S_{N_{IS}}$ tend vers S (définie dans l'énoncé du théorème) lorsque N_{IS} tend vers l'infini. De plus, C_1 et C_3 dépendent de manière continue à N_{IS}. On peut donc majorer uniformément ces deux constantes dans le voisinage de la matrice cible. Appelons $\mathcal{C}_1(d)$ et $\mathcal{C}_3(d)$ ces majorants.

Ceci implique donc pour $A^{N_{IS}}(d, N_{IS})$:

$$A^{N_{IS}}(d, N_{IS}) \leqslant \frac{\mathcal{C}_1(d)}{\sqrt{N_{IS}}} \tag{10.5}$$

À présent, intéressons nous au terme $B^{N_{IS}}(d, N_{IS})$. $\mathcal{B}(c, r)$ est la boule de centre c et de rayon r, $\omega_g(A) = \sup_{t,y\in A} |g(t) - g(y)|$ et A une partie de $\mathbb{R}^d$. Or dans notre cas, $g = I_u(y) = I(y + u)$ avec I la fonction indicatrice $\mathbf{1}$. Ceci signifie que :

$$\forall v \in \mathbb{R}^d \qquad \omega_{I_u}\left(\mathcal{B}\left(v, \frac{C_3(d, N_{IS})}{\sqrt{N_{IS}}}\right)\right) \leqslant \omega_{I_u}\left(\mathcal{B}\left(v, \frac{\mathcal{C}_3(d)}{\sqrt{N_{IS}}}\right)\right) \leqslant 1$$

Nous avons donc :

$$B^{N_{IS}}(d, N_{IS}) \leqslant k(\Phi) \int_{z\in\mathcal{B}\left(v+u, \frac{\mathcal{C}_3(d)}{\sqrt{N_{IS}}}\right)} dz = \underbrace{C_4(d, \Phi) N_{IS}^{-d/2}}_{\text{volume de la boule}} = o\left(\frac{1}{\sqrt{N_{IS}}}\right) \tag{10.6}$$

La dernière égalité dans l'équation (10.6) se justifie par le fait que $d \geqslant 2$, par hypothèse.

En repartant de l'équation (10.4) et par les majorations fournies par les équations (10.5) et (10.6), on en déduit que quand $N_{IS} \to \infty$ et $\forall \boldsymbol{v} \in \mathbb{R}^d$, on a :

$$\left| \tilde{\mathbb{P}}\left[\bigcap_{k=1}^{d} \left\{\psi_{N_{IS}}^k(t) \leqslant v^k\right\}\right] - \tilde{\mathbb{P}}\left[\bigcap_{k=1}^{d} \left\{\phi_k^{N_{IS}} \leqslant v^k\right\}\right] \right| \underset{N_{IS}\to\infty}{\to} 0$$

□

En revenant à l'équation (10.3) et en appliquant le lemme précédent, on a :

$$\tilde{\mathbb{P}}\left[\bigcap_{k=1}^{d} \left\{\alpha \leqslant \tilde{H}_{N_{IS}}(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k)\right\}\right] = \tilde{\mathbb{P}}\left[\bigcap_{k=1}^{d} \left\{\phi_k^{N_{IS}} \geqslant a_{N_{IS}}^k(t)\right\}\right] + O\left(\frac{1}{\sqrt{N_{IS}}}\right)$$

Maintenant pour le comportement de $a_{N_{IS}}^k(t)$ quand $N_{IS} \to \infty$, on peut écrire :

$$\begin{array}{rcl} a_{N_{IS}}^k(t) & = & \sqrt{N_{IS}}\left[\alpha - H\left(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k\right)\right] \\ & = & t^k \dfrac{H\left(q_\alpha(\boldsymbol{x}^k);\ \boldsymbol{x}^k\right) - H\left(q_\alpha(\boldsymbol{x}^k) + \frac{t^k}{\sqrt{N_{IS}}};\ \boldsymbol{x}^k\right)}{\frac{t^k}{\sqrt{N_{IS}}}} \\ & \underset{N_{IS}\to\infty}{=} & -t^k H'\left(q_\alpha(\boldsymbol{x}^k);\ \boldsymbol{x}^k\right)) \end{array}$$

De plus, lorsque N_{IS} tend vers l'infini, on a $\boldsymbol{\phi}^{N_{IS}} \xrightarrow{\mathcal{L}} \boldsymbol{\phi}$ avec $\boldsymbol{\phi} \sim \mathcal{N}(0, S)$ avec S défini dans l'énoncé du théorème. En reprenant l'équation (10.3) lorsque N_{IS} tend vers l'infini, nous obtenons :

$$\begin{aligned} &\tilde{\mathbb{P}}\left[\sqrt{N_{IS}}(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}^1) - q_\alpha(\boldsymbol{x}^1)) \leqslant t_1 \cap \cdots \cap \sqrt{N_{IS}}(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}^j) - q_\alpha(\boldsymbol{x}^j)) \leqslant t_j\right] \\ &= \tilde{\mathbb{P}}\left(\phi_1 < t^1 H'(q_\alpha(\boldsymbol{x}^1);\ \boldsymbol{x}^1) \cap \cdots \cap \phi_d < t^d H'(q_\alpha(\boldsymbol{x}^d);\ \boldsymbol{x}^d)\right) + O\left(\frac{1}{\sqrt{N_{IS}}}\right) \end{aligned} \tag{10.7}$$

Ce qui est équivalent au résultat annoncé.

□

Cas du superquantile : Afin de regarder la corrélation de l'erreur en d différents points d'optimisation $\left(\boldsymbol{x}^k\right)_{1\leqslant k\leqslant d}$, il faut redéfinir la fonction $\Psi^{N_{IS}}$ définie dans l'étude [Rockafellar et Uryasev, 2000].

$$\Psi^{N_{IS}}(\boldsymbol{x},\gamma) = \left[\gamma + \frac{1}{(1-\alpha)\,N_{IS}} \sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i)\mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\geqslant\gamma}\left(f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)-\gamma\right)\right] \tag{10.8}$$

Pour rappel, le superquantile $\tilde{Q}_\alpha^{IS}(\boldsymbol{x})$ vérifie $\tilde{Q}_\alpha^{IS}(\boldsymbol{x}) = \min_{\gamma\in\mathbb{R}}\Psi^{N_{IS}}(\boldsymbol{x},\gamma)$. Toutefois, de sorte à rester le plus général possible, le théorème est démontré dans le cas où γ varie également, c'est-à-dire que l'erreur est estimée en d différents couples $(\gamma,x) : (\gamma_1,\boldsymbol{x}^1),\cdots,(\gamma_d,\boldsymbol{x}^d)$.

Théorème 10.3. *Si* $\forall k\in\{1\cdots d\}$, $\forall x\in\mathcal{D}_x$ *et* $\forall\gamma\in\mathbb{R}$, $\mathbb{E}[L\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^k)^2\left(f(x,\tilde{\boldsymbol{\xi}}^1)-\gamma\right)^2] < \infty$, *alors :*

$$\sqrt{N_{IS}}\left[\begin{pmatrix}\Psi^{N_{IS}}(\boldsymbol{x}^1,\gamma_1)\\ \cdots\\ \Psi^{N_{IS}}(\boldsymbol{x}^d,\gamma_d)\end{pmatrix} - \begin{pmatrix}\Psi(\boldsymbol{x}^1,\gamma_1)\\ \cdots\\ \Psi(\boldsymbol{x}^d,\gamma_d)\end{pmatrix}\right] \xrightarrow[N_{IS}\to\infty]{\mathcal{L}} \mathcal{N}(0,\Sigma)$$

Où les différentes quantités sont définies telles que :

$$\Sigma_{ij} = \frac{1}{(1-\alpha)^2}\begin{pmatrix}\mathbb{E}\left[\mathbf{1}_{f(\boldsymbol{x}^i,\tilde{\boldsymbol{\xi}}^1)\geqslant\gamma_i}\mathbf{1}_{f(\boldsymbol{x}^j,\tilde{\boldsymbol{\xi}}^1)\geqslant\gamma_j}\left(f(\boldsymbol{x}^i,\tilde{\boldsymbol{\xi}}^1)-\gamma_i\right)\left(f(\boldsymbol{x}^j,\tilde{\boldsymbol{\xi}}^1)-\gamma_j\right)L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^j)L(\tilde{\boldsymbol{\xi}}^1;\ \boldsymbol{x}^i)\right] -\\ \int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}}\mathbf{1}_{f(\boldsymbol{x}^i,\boldsymbol{\chi})\geqslant\gamma_i}\left(f(\boldsymbol{x}^i,\boldsymbol{\chi})-\gamma_i\right)L(\boldsymbol{\chi};\ \boldsymbol{x}^i)h_b(\boldsymbol{\chi})\,d\boldsymbol{\chi}\times\\ \int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}}\mathbf{1}_{f(\boldsymbol{x}^j,\boldsymbol{\chi})\geqslant\gamma_j}\left(f(\boldsymbol{x}^j,\boldsymbol{\chi})-\gamma_j\right)L(\boldsymbol{\chi};\ \boldsymbol{x}^j)h_b(\boldsymbol{\chi})\,d\boldsymbol{\chi}\end{pmatrix}$$

Preuve.

Posons $\boldsymbol{U}_i = \begin{pmatrix} U_i^{\gamma_1,\boldsymbol{x}^1} = \frac{\mathbf{1}_{f(\boldsymbol{x}^1,\tilde{\boldsymbol{\xi}}^i)\geqslant\gamma_1}\left(f(\boldsymbol{x}^1,\tilde{\boldsymbol{\xi}}^i)-\gamma_1\right)L(\tilde{\boldsymbol{\xi}}^i);\ \boldsymbol{x}^1}{1-\alpha}\\ \cdots\\ U_i^{\gamma_d,\boldsymbol{x}^d} = \frac{\mathbf{1}_{f(\boldsymbol{x}^d,\tilde{\boldsymbol{\xi}}^i)\geqslant\gamma_d}\left(f(\boldsymbol{x}^d,\tilde{\boldsymbol{\xi}}^i)-\gamma_d\right)L(\tilde{\boldsymbol{\xi}}^i);\ \boldsymbol{x}^d}{1-\alpha}\end{pmatrix}$. Alors, $U_1,\cdots,U_{N_{IS}}$ est une suite de variables i.i.d de matrice de covariance Σ. Ceci implique donc que l'espérance $\mathbb{E}(\boldsymbol{U}_i)$ vaut donc :

$$\mathbb{E}(\boldsymbol{U}_i) = \begin{pmatrix}\frac{1}{1-\alpha}\int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}}\mathbf{1}_{f(\boldsymbol{x}^1,\boldsymbol{\chi})\geqslant\gamma_1}\left(f(\boldsymbol{x}^1,\boldsymbol{\chi})-\gamma_1\right)L(\boldsymbol{\chi};\ \boldsymbol{x}^1)h_b(\boldsymbol{\chi})\,d\boldsymbol{\chi}\\ \cdots\\ \frac{1}{1-\alpha}\int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}}\mathbf{1}_{f(\boldsymbol{x}^d,\boldsymbol{\chi})\geqslant\gamma_d}\left(f(\boldsymbol{x}^d,\boldsymbol{\chi})-\gamma_d\right)L(\boldsymbol{\chi};\ \boldsymbol{x}^d)h_b(\boldsymbol{\chi})\,d\boldsymbol{\chi}\end{pmatrix}$$

On a donc :

$$\left[\begin{pmatrix}\Psi^{N_{IS}}(\boldsymbol{x}^1,\gamma_1)\\ \cdots\\ \Psi^{N_{IS}}(\boldsymbol{x}^d,\gamma_d)\end{pmatrix} - \begin{pmatrix}\Psi(\boldsymbol{x}^1,\gamma_1)\\ \cdots\\ \Psi(\boldsymbol{x}^d,\gamma_d)\end{pmatrix}\right] = \frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}}\boldsymbol{U}_i - \mathbb{E}(\boldsymbol{U}_i)$$

Par application directe du Théorème de la limite centrale sur la variable $\boldsymbol{U}_i$ [Saporta, 2006, p.92], on obtient le résultat annoncé. □

10.3 Preuve de la convergence de l'erreur vers un processus aléatoire

On cherche à prouver que la fonction $\left(\rho^{N_{IS}}(\boldsymbol{x})\right)_{\boldsymbol{x}\in\mathcal{D}_x}$ converge, aussi bien pour le quantile que pour le superquantile. On s'intéresse donc à la convergence d'une suite de variables aléatoires $(U_n)_{n\in\mathbb{N}}$ à valeurs dans $\Omega = \mathcal{C}^0(\mathcal{D}_x, \mathbb{R})$ et de mesure de probabilité μ_n. D'après [Billingsley, 2008], la convergence se définit comme suit :

Définition 10.1. Une suite de variables aléatoires $(U_n)_{n\in\mathbb{N}}$ de mesure de probabilité μ_n converge vers une variable aléatoire U de mesure de probabilité μ si $\forall f : \Omega \longmapsto \mathbb{R}$ continue bornée, $\mathbb{E}[f(U_n)] \to \mathbb{E}[f(U)]$.

De manière équivalente, la convergence peut se démontrer grâce au théorème suivant [Billingsley, 2008] :

Théorème 10.4. *Une suite de variables aléatoires $(U_n)_{n\in\mathbb{N}}$ de mesure de probabilité μ_n converge vers une variable aléatoire U de mesure de probabilité μ si les deux conditions suivantes sont vérifiées :*

1. *La suite U_n est relativement compacte, ce qui permet de dire qu'il existe des sous suites convergentes.*
2. *Il y a unicité des valeurs d'adhérence, c'est-à-dire que toutes les sous-suites convergentes de $(U_n)_{n\in\mathbb{N}}$ convergent en loi vers U*

Alors $(U_n)_{n\in\mathbb{N}}$ converge vers U.

Dans le cas du quantile et du superquantile, il faut donc démontrer ces deux propriétés.

Preuve de la relative compacité Pour prouver la compacité demandée au point 1 du Théorème 10.4, il est fréquent de faire appel au concept de tension d'une famille de mesure puisque les deux concepts sont liés par le théorème de Prokhorov [Prokhorov, 1956]. La tension d'une suite de mesures est définie comme suit :

Définition 10.2. Une famille de mesure $\mathcal{M}$ est tendue si $\forall\varepsilon > 0$, $\exists K \subset \Omega$ tel que K est compact et $\mu(K) > 1 - \varepsilon\ \forall\mu \in \mathcal{M}$.

Maintenant le théorème de Prokhorov permet de faire le lien entre tension et relative compacité :

Théorème 10.5 (Théorème de Prokhorov)**.** *Si la famille de mesure $\mathcal{M}$ est tendue, alors elle est relativement compacte*

Pour mettre en évidence la relative compacité de la suite des fonctions quantile et superquantile, nous devons donc montrer que ces suites sont tendues.

10.3.1 Cas du quantile

Afin de rendre la preuve de la tension possible, il est nécessaire de travailler sur des fonctions suffisamment régulières. Pour cela, la fonction de répartition empirique

$\tilde{H}_{N_{IS}}(y;\ \boldsymbol{x})$ doit être rendue continue, différentiable. Introduisons donc la fonction sigmoïde $\phi_\lambda(u) = \frac{1}{1+\exp(-\lambda u)}$ avec $\lambda > 0$. Soit $\tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x})$ la version continue de $\tilde{H}_{N_{IS}}(y;\ \boldsymbol{x})$:

$$\tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x}) = \frac{1}{N_{IS}} \sum_{i=1}^{N_{IS}} \phi_\lambda \left(y - f(\boldsymbol{x}, \tilde{\boldsymbol{\xi}}^i) \right) L(\tilde{\boldsymbol{\xi}}^i)$$

On cherche donc à prouver que $\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})$ converge vers $q_{\alpha,\lambda}(\boldsymbol{x})$, une valeur déterministe avec :

$$\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}) = \inf \left\{ \ y \ : \tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x}) \geqslant \alpha \right\}$$

La suite de processus aléatoires dont nous souhaitons prouver la tension est donc la suivante :

$$\begin{array}{rcl} \mathcal{D}_x & \longmapsto & \mathbb{R} \\ \boldsymbol{x} & \longrightarrow & G^{N_{IS}}(\boldsymbol{x}) = \tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}) \end{array}$$

Dans cette partie, on fait l'hypothèse que $L(\tilde{\boldsymbol{\xi}}^j)$ ne dépend pas de x. Ceci permet notamment de simplifier les calculs. De plus, c'est le cas sur tout le domaine $\boldsymbol{x}$ en dehors des points du bord.

Lemme 10.4. *Si $\lambda > 0$ et $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$ avec $\mathcal{D}_x$ et $\mathcal{D}_\xi$ deux ensembles bornés.*
Alors la suite $\left(\tilde{q}_{\alpha,\lambda}^{N_{IS}}\right)_{N_{IS}\in\mathbb{N}}$ est tendue.

Preuve.
Pour montrer la tension de cette famille, d'après l'étude de Kunita [Kunita, 1997, p.38], il faut montrer que $\forall N_{IS}$, $\exists \gamma > 0$, $\exists C > 0$ et $\exists \alpha_i > 0$ avec $\sum_{i=1}^d \alpha_i^{-1} < 1$ tel que :

$$\mathbb{E}\left[\left| \tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}) - \tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}') \right|^\gamma \right] \leqslant C \sum_{i=1}^d \left| \boldsymbol{x}^i - \boldsymbol{x}^{i\prime} \right|^{\alpha_i} \quad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{D}_x \subset \mathbb{R}^{N_x} \text{ (borné ici)}$$

Si $G^{N_{IS}}$ est suffisamment différentiable (ici $\mathcal{C}^1$ suffit), étant donné que l'on travaille sur des espaces d'entrée bornés, par le théorème des accroissements finis multi-dimensionnel et par l'utilisation de la norme $N_x + 1$, on a :

$$\mathbb{E}\left[\left| \tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}) - \tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}') \right|^{N_x+1} \right] \leqslant \mathbb{E}\left[\left\| \sup_{\boldsymbol{x}} \text{grad} G^{N_{IS}} \right\|^{N_x+1} \right] \left\| \boldsymbol{x} - \boldsymbol{x}' \right\|^{N_x+1} \tag{10.9}$$

Or, $\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})$ est solution de l'équation suivante $\forall \boldsymbol{x}$, $\tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x}) = \alpha$. De plus, sachant que la fonction $\tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x})$ est croissante par rapport à y à $\boldsymbol{x}$ fixé, il y a unicité de la solution $\forall \boldsymbol{x} \in \mathcal{D}_x$.

Pour en déduire de cette équation une formule pour le gradient de $G^{N_{IS}}$, il est donc possible d'utiliser le théorème des fonctions implicites. En effet, on peut voir cette équation de la manière suivante en introduisant $\varphi(y;\ \boldsymbol{x})$:

$$\varphi(y;\ \boldsymbol{x}) = \frac{1}{N_{IS}} \sum_{i=1} \phi_\lambda \left(y - f(\boldsymbol{x}, \tilde{\boldsymbol{\xi}}^i) \right) L(\tilde{\boldsymbol{\xi}}^i) - \alpha = 0$$

Puisque $f \in \mathcal{C}^1$ et que

$$\left.\frac{\partial \varphi(y;\ \boldsymbol{x})}{\partial y}\right|_{\substack{y=\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}^0) \\ \boldsymbol{x}=\boldsymbol{x}^0}} = \left.\frac{\partial \tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x})}{\partial y}\right|_{\substack{y=\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}^0) \\ \boldsymbol{x}=\boldsymbol{x}^0}} = \frac{1}{N_{IS}} \sum_{i=1} \frac{\lambda \exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}} - f(\boldsymbol{x}, \tilde{\boldsymbol{\xi}}^i)))}{(1 + \exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}} - f(\boldsymbol{x}, \tilde{\boldsymbol{\xi}}^i))))^2} L(\tilde{\boldsymbol{\xi}}^i) > 0$$

Le théorème des fonctions implicites est applicable sur $\varphi(y;\ \boldsymbol{x}) = 0$, $\forall \boldsymbol{x}^0 \in \mathcal{D}_x$. Il est ainsi possible de calculer le gradient de la solution $G^{N_{IS}}(\boldsymbol{x})$ en tous les points $\boldsymbol{x}^0$ vérifiant cette inégalité (donc tous ceux de l'ensemble de définition de $G^{N_{IS}}$) et on a pour toutes les composantes $1 \leqslant j \leqslant N_x$:

$$\left.\frac{\partial G^{N_{IS}}(\boldsymbol{x})}{\partial x_j}\right|_{\boldsymbol{x}=\boldsymbol{x}^0} = -\left.\frac{\partial\varphi(y;\ \boldsymbol{x})/\partial x_j}{\partial\varphi(y;\ \boldsymbol{x})/\partial y}\right|_{\substack{y=\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}^0)\\ \boldsymbol{x}=\boldsymbol{x}^0}} = \left.\frac{\partial\tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x})/\partial x_j}{\partial\tilde{H}_{N_{IS},\lambda}(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}^0);\ \boldsymbol{x}^0)/\partial y}\right|_{\substack{y=\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x}^0)\\ \boldsymbol{x}=\boldsymbol{x}^0}}$$

Or

$$\frac{\partial\tilde{H}_{N_{IS},\lambda}(y);\ \boldsymbol{x})}{\partial x_j} = \frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}}\frac{\lambda\exp(-\lambda(y-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)))}{(1+\exp(-\lambda(y-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i))))^2}\frac{\partial f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)}{\partial x_j}L(\tilde{\boldsymbol{\xi}}^i)$$

et

$$\frac{\partial\tilde{H}_{N_{IS},\lambda}(y;\ \boldsymbol{x})}{\partial y} = \frac{1}{N_{IS}}\sum_{i=1}^{N_{IS}}\frac{\lambda\exp(-\lambda(y-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)))}{(1+\exp(-\lambda(y-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i))))^2}L(\tilde{\boldsymbol{\xi}}^i)$$

Ce qui donne, $\forall \boldsymbol{x} \in \mathcal{D}_x$ (un borné de $\mathbb{R}^{N_x}$) :

$$\left.\frac{\partial G^{N_{IS}}(\boldsymbol{x})}{\partial x_j}\right|_{\boldsymbol{x}=\boldsymbol{x}^0} = \frac{\sum_{i=1}^{N_{IS}}\frac{\exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)))}{(1+\exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i))))^2}\frac{\partial f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)}{\partial x_j}L(\tilde{\boldsymbol{\xi}}^i)}{\sum_{i=1}^{N_{IS}}\frac{\exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)))}{(1+\exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i))))^2}L(\tilde{\boldsymbol{\xi}}^i)}$$

qui est en fait une moyenne pondérée par $\frac{\exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)))}{(1+\exp(-\lambda(\tilde{q}_{\alpha,\lambda}^{N_{IS}}(\boldsymbol{x})-f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i))))^2}L(\tilde{\boldsymbol{\xi}}^i)$ des $\frac{\partial f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)}{\partial x_j}$.

Puisque $\mathcal{D}_x$ et $\mathcal{D}_\xi$ sont bornés et que $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$, l'équation (10.9) est applicable et il est tout à fait facile de borner cette quantité et d'obtenir l'inégalité recherchée avec $\forall N_{IS}$, $\gamma = N_x + 1 > 0$, $C = \left\|\sup_y \text{grad}G^{N_{IS}}(y)\right\|^{N_x+1} > 0$ et $\alpha_i = N_x + 1$ (donc $\sum_{i=1}^d \alpha_i^{-1} = \frac{d}{N_x+1} < 1$). □

Maintenant que la relative compacité a été démontrée pour la version modifiée du quantile, il est nécessaire de prouver que cette version modifiée converge en tout point $\boldsymbol{x} \in \mathcal{D}_x$. Ceci permet ainsi de prouver l'unicité de la valeur d'adhérence puisqu'elle est déterministe.

Lemme 10.5. *Si $\mathbb{E}[L(\tilde{\boldsymbol{\xi}}^1)^3] < \infty$ et la distribution de la sortie H_λ est différentiable en $q_{\alpha,\lambda} = H_\lambda^{-1}(\alpha)$ avec $H'_\lambda(q_{\alpha,\lambda}) = h_{\lambda,Y}(q_{\alpha,\lambda}) > 0$, alors :*

$$\sqrt{N_{IS}}\left(\tilde{q}_{\alpha,\lambda}^{IS} - q_{\alpha,\lambda}\right) \underset{N_{IS}\to\infty}{\xrightarrow{\mathcal{L}}} \sigma_{q,\lambda,IS}\,\mathcal{N}(0,1) \text{ avec } \sigma^2_{q,\lambda,IS} = \frac{\text{Var}\left(\phi_\lambda\left(q_{\alpha,\lambda} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)L(\tilde{\boldsymbol{\xi}}^1)\right)}{h_{Y,\lambda}(q_{\alpha,\lambda})^2} \tag{10.10}$$

Preuve.
Montrons que $\forall N_{IS} \in \mathbb{N}^\star$, $\tilde{q}_{\alpha,\lambda}^{IS} \leqslant t \Leftrightarrow \tilde{H}_{N_{IS},\lambda}(t) \geqslant \alpha$. Prenons t tel que $\tilde{q}_{\alpha,\lambda}^{IS} \leqslant t$. Alors par définition de $\tilde{q}_{\alpha,\lambda}^{IS}$ (étant l'inf de l'ensemble $\left\{t\ :\ \tilde{H}_{N_{IS},\lambda}(t) \geqslant \alpha\right\}$), on a $\tilde{H}_{N_{IS},\lambda}(t) \geqslant \alpha$. Dans l'autre sens, si t est tel que $\tilde{H}_{N_{IS},\lambda}(t) \geqslant \alpha$, alors par définition de l'inf, on a encore

$\tilde{q}^{IS}_{\alpha,\lambda} \leqslant t$. Ce qui montre que ces deux inégalités sont équivalentes.

On peut donc écrire :

$$\tilde{\mathbb{P}}\left[\sqrt{N_{IS}}(\tilde{q}^{IS}_{\alpha,\lambda} - q_{\alpha,\lambda}) \leqslant t\right] = \tilde{\mathbb{P}}\left[\tilde{q}^{IS}_{\alpha,\lambda} \leqslant q_{\alpha,\lambda} + \frac{t}{\sqrt{N_{IS}}}\right] = \tilde{\mathbb{P}}\left[\alpha \leqslant \tilde{H}_{N_{IS},\lambda}\left(q_{\alpha,\lambda} + \frac{t}{\sqrt{N_{IS}}}\right)\right] \tag{10.11}$$

Introduisons les notations suivantes

$$\left\{\begin{array}{rcl} s_{N_{IS}}(t) & = & \left\{\tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^1)^2\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\right)\right] - H^2_\lambda(q_{\alpha,\lambda} + t/\sqrt{N_{IS}})\right\}^{1/2} \\ \psi_{N_{IS}}(t) & = & \dfrac{\sqrt{N_{IS}}}{s_{N_{IS}}(t)}\left[\tilde{H}_{N_{IS},\lambda}\left(q_{\alpha,\lambda} + \dfrac{t}{\sqrt{N_{IS}}}\right) - H_\lambda\left(q_{\alpha,\lambda} + \dfrac{t}{\sqrt{N_{IS}}}\right)\right] \\ a_{N_{IS}}(t) & = & \dfrac{\sqrt{N_{IS}}}{s_{N_{IS}}(t)}\left[\alpha - H_\lambda\left(q_{\alpha,\lambda} + \dfrac{t}{\sqrt{N_{IS}}}\right)\right] \end{array}\right.$$

Avec ces notations, on a donc :

$$\tilde{\mathbb{P}}\left[\alpha \leqslant \tilde{H}_{N_{IS},\lambda}\left(q_{\alpha,\lambda} + \frac{t}{\sqrt{N_{IS}}}\right)\right] = \tilde{\mathbb{P}}\left[a_{N_{IS}}(t) \leqslant \psi_{N_{IS}}(t)\right] \underbrace{-\tilde{\mathbb{P}}\left[\mathcal{N}(0,1) \geqslant a_{N_{IS}}(t)\right] + \tilde{\mathbb{P}}\left[\mathcal{N}(0,1) \geqslant a_{N_{IS}}(t)\right]}_{=0!}$$

On peut ainsi réécrire $\psi_{N_{IS}}$ de la manière suivante :

$$\psi_{N_{IS}}(t) = \frac{\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\right) - \tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^1)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)\right]}{\sqrt{N_{IS}}\ \left(\mathrm{Var}\left[L(\tilde{\boldsymbol{\xi}}^1)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)\right]\right)^{1/2}}$$

$$\text{Car } \left\{\begin{array}{l} \tilde{H}_{N_{IS},\lambda}\left(q_{\alpha,\lambda} + \dfrac{t}{\sqrt{N_{IS}}}\right) = \dfrac{1}{N_{IS}}\displaystyle\sum_{i=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^i)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^i)\right) \\ H_\lambda(q_{\alpha,\lambda} + t/\sqrt{N_{IS}}) = \tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^1)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)\right] \\ s^2_{N_{IS}}(t) = \mathrm{Var}\left[L(\tilde{\boldsymbol{\xi}}^1)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)\right] \end{array}\right.$$

En appliquant le théorème de Berry-Esseen [Berry, 1941], on a donc l'inégalité suivante :

$$\sup_t \left|\tilde{\mathbb{P}}\left[\psi_{N_{IS}}(t) \leqslant x\right] - \Phi_{(0,1)}(x)\right| \leqslant C\frac{\tilde{\rho}}{\tilde{\sigma}^3\sqrt{N_{IS}}}$$

Ici, $\Phi_{(0,1)}$ correspond à la fonction de répartition de la loi normale (définie à l'annexe A), $\tilde{\sigma} = \tilde{\mathbb{E}}\left(\left[L(\tilde{\boldsymbol{\xi}}^1)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)\right]^2\right)$ et
$\tilde{\rho} = \tilde{\mathbb{E}}\left(\left[L(\tilde{\boldsymbol{\xi}}^1)\ \phi_\lambda\left(q_{\alpha,\lambda} + t/\sqrt{N_{IS}} - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^1)\right)\right]^3\right)$.
Or lorsque $N_{IS} \to \infty$, $\tilde{\rho}$ et $\tilde{\sigma}$ tendent vers une limite finie, grâce aux hypothèses de régularité sur $L(\tilde{\boldsymbol{\xi}}^1)$ du théorème et au fait que ϕ_λ est bornée par 1. Ce qui signifie que $C\frac{\tilde{\rho}}{\tilde{\sigma}^3\sqrt{N_{IS}}} \underset{N_{IS}\infty}{\to} 0$. En repartant de l'équation (10.11), on a donc

$$\tilde{\mathbb{P}}\left(\sqrt{N_{IS}}(\tilde{q}^{IS}_{\alpha,\lambda} - q_{\alpha,\lambda}) \leqslant t\right) \underset{N_{IS}\infty}{=} \Phi_{(0,1)}\left(a_{N_{IS}}(t)\right)$$

Et pour le comportement de $a_{N_{IS}}(t)$ quand $N_{IS} \to \infty$, on peut écrire :

$$\begin{aligned} a_{N_{IS}}(t) &= \frac{\sqrt{N_{IS}}}{s_{N_{IS}}(t)}\left[\alpha - H_\lambda\left(q_{\alpha,\lambda} + \frac{t}{\sqrt{N_{IS}}}\right)\right] \\ &= \frac{t}{s_{N_{IS}}(t)} \frac{H_\lambda(q_{\alpha,\lambda}) - H_\lambda\left(q_{\alpha,\lambda} + \frac{t}{\sqrt{N_{IS}}}\right)}{\frac{t}{\sqrt{N_{IS}}}} \\ &\underset{N_{IS}\to\infty}{=} -\frac{t}{s} H'_\lambda(q_{\alpha,\lambda}) = -\frac{t}{\sigma_{q,IS}} \end{aligned}$$

Ce qui nous donne en revenant à l'équation (10.11) pour N_{IS} tendant vers ∞ :

$$\begin{aligned} \tilde{\mathbb{P}}\left(\sqrt{N_{IS}}(\tilde{q}^{IS}_{\alpha,\lambda} - q_{\alpha,\lambda}) \leqslant t\right) &= \tilde{\mathbb{P}}\left(\mathcal{N}(0,1) \geqslant -\frac{t}{\sigma_{q,IS}}\right) + O\left(\frac{1}{\sqrt{N_{IS}}}\right) \\ &= \tilde{\mathbb{P}}\left(\sigma_{q,IS}\mathcal{N}(0,1) \leqslant t\right) + O\left(\frac{1}{\sqrt{N_{IS}}}\right) \end{aligned} \tag{10.12}$$

□

On a donc montré que la version modifiée du quantile converge $\forall \boldsymbol{x} \in \mathcal{D}_x$. Nous pouvons donc énoncer le théorème de convergence suivant :

Théorème 10.6. *Si $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$ avec $\mathcal{D}_x$ et $\mathcal{D}_\xi$ deux sous-espaces bornés. alors $\tilde{q}^{N_{IS}}_{\alpha,\lambda}$ converge vers une fonction déterministe définie dans $\mathcal{C}^1(\mathcal{D}_x, \mathbb{R})$.*

Preuve.
Pour prouver la convergence, d'après le théorème 10.4, il est suffisant de montrer qu'il y a unicité de la valeur d'adhérence et que la suite est relativement compacte.

Pour la preuve de la compacité, on prouve d'abord la tension de la suite $\tilde{q}^{N_{IS}}_{\alpha,\lambda}$ au théorème 10.4. Puis par application du théorème de Prokhorov (théorème 10.5), on obtient la relative compacité.

Pour l'unicité, le lemme 10.5 prouve la convergence de ce processus $\forall \boldsymbol{x} \in \mathcal{D}_x$. Montrons à présent que cela suffit à prouver l'unicité de la valeur d'adhérence. Ceci se montre par l'absurde.

Supposons qu'il n'existe pas une unique valeur d'adhérence. Ceci signifie qu'il existe au moins deux valeurs d'adhérence $g_1(\boldsymbol{x})$ et $g_2(\boldsymbol{x})$ supposées distinctes. Ceci signifie notamment qu'il existe un $\boldsymbol{x} \in \mathcal{D}_x$ tel que $g_1(\boldsymbol{x}) \neq g_2(\boldsymbol{x})$. Or, par le lemme 10.5, $\tilde{q}^{N_{IS}}_{\alpha,\lambda}$ tend vers une valeur déterministe donnée $q_{\alpha,\lambda}(\boldsymbol{x})$ pour chaque $\boldsymbol{x} \in \mathcal{D}_x$. Il existe donc une unique valeur possible pour la convergence en chaque point $\mathcal{D}_x$.

Ceci contredit l'hypothèse qu'il existe deux valeurs d'adhérence. Il ne peut donc en exister qu'une seule. □

10.3.2 Cas du superquantile

On cherche à prouver que $\left(\Psi^{N_{IS}}(\boldsymbol{x},\gamma)\right)$ tend vers une fonction de $(\boldsymbol{x},\gamma) \in \mathcal{D}_x \times \mathbb{R}$ déterministe.

En supposant que $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi)$, on a $\Psi^{N_{IS}} \in \Omega = \mathcal{C}^0(\mathcal{D}_x \times \mathbb{R}, \mathbb{R})$

Lemme 10.6. *Si $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$ avec $\mathcal{D}_x$ et $\mathcal{D}_\xi$ deux compacts de $\mathbb{R}^{N_x}$ et de $\mathbb{R}^{N_\xi}$ respectivement.*

Alors la suite $\left(\Psi^{N_{IS}}(\boldsymbol{x},\gamma)\right)_{N_{IS}\in\mathbb{N}}$ est tendue.

Preuve.
Pour montrer la tension de cette suite, d'après l'étude de Billingsley [Billingsley, 2008, p.82], il faut montrer que $\forall\varepsilon > 0$, $\forall\eta > 0$, $\exists\delta > 0$, $\exists n_0 \in \mathbb{N}^\star$ tel que :

$$\mathbb{P}\left[\sup_{\|(\boldsymbol{x},\gamma)-(\boldsymbol{x}',\gamma')\|\leqslant\delta} \left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right| > \varepsilon\right] \leqslant \eta, \quad \forall N_{IS} \geqslant n_0$$

Regardons d'abord ce que vaut la différence $\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right|$:

$$\Psi^{N_{IS}}(\boldsymbol{x},\gamma)-\Psi^{N_{IS}}(\boldsymbol{x}',\gamma') = \frac{1}{N_{IS}}\sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\left\{\left[f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j)-\gamma\right]\mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j)\geqslant\gamma} - \left[f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j)-\gamma'\right]\mathbf{1}_{f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j)\geqslant\gamma'}\right\}$$

Maintenant, posons :

$$\left\{\begin{array}{rcl} I_1 &=& \left\{i \ : \ f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) > \gamma \cap f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) > \gamma'\right\} \\ I_2 &=& \left\{i \ : \ f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) > \gamma \cap f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) \leqslant \gamma'\right\} \\ I_3 &=& \left\{i \ : \ f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) \leqslant \gamma \cap f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) > \gamma'\right\} \end{array}\right.$$

Grâce à ces notations, nous pouvons écrire :

$$\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma') = \frac{1}{N_{IS}}\sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\left\{\mathcal{S}_1^j\mathbf{1}_{j\in I_1} + \mathcal{S}_2^j\mathbf{1}_{j\in I_2} + \mathcal{S}_3^j\mathbf{1}_{j\in I_3}\right\} \tag{10.13}$$

avec :

$$\left\{\begin{array}{rcl} \mathcal{S}_1^j &=& f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) + \gamma' - \gamma \\ \mathcal{S}_2^j &=& f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - \gamma \\ \mathcal{S}_3^j &=& f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) - \gamma' \end{array}\right.$$

Or, lorsque $j \in I_2$, par définition de I_2 on a $\gamma' - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) \geqslant 0$ et donc :

$$\mathcal{S}_2^j\mathbf{1}_{j\in I_2} \leqslant \left(f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) + \gamma' - \gamma\right)\mathbf{1}_{j\in I_2}$$

De la même manière, lorsque $j \in I_3$, on a :

$$\mathcal{S}_3^j\mathbf{1}_{j\in I_3} \leqslant \left(f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) + \gamma - \gamma'\right)\mathbf{1}_{j\in I_3}$$

En revenant à l'équation (10.13), on a donc :

$$\Psi^{N_{IS}}(\boldsymbol{x},\gamma)-\Psi^{N_{IS}}(\boldsymbol{x}',\gamma') \leqslant \frac{1}{N_{IS}}\sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\left(f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) + \gamma' - \gamma\right)\{\mathbf{1}_{j\in I_1} + \mathbf{1}_{j\in I_2} - \mathbf{1}_{j\in I_3}\}$$

Or, $I_1 \cap I_2 = \emptyset$, $I_1 \cap I_3 = \emptyset$ et $I_2 \cap I_3 = \emptyset$. Ceci implique que $\{\mathbf{1}_{j\in I_1} + \mathbf{1}_{j\in I_2} - \mathbf{1}_{j\in I_3}\} \leqslant 1$ pour tout $j \in \{1, \cdots, N_{IS}\}$. On a donc :

$$\begin{aligned}\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right| &\leqslant \frac{1}{N_{IS}}\sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\left|f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j) + \gamma' - \gamma\right| \\ &\leqslant \frac{1}{N_{IS}}\sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j)\left|f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j)\right| + |\gamma' - \gamma|\end{aligned}$$

En nous plaçons dans le cas où $\|(\boldsymbol{x},\gamma) - (\boldsymbol{x}',\gamma')\| \leqslant \delta$, on a $|\gamma' - \gamma| \leqslant \delta$. Enfin, afin de borner la partie $\left|f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j)\right|$ à l'aide de δ, nous pouvons utiliser les hypothèses sur f de l'énoncé du lemme. En effet, $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$ avec $\mathcal{D}_x$ et $\mathcal{D}_\xi$ deux compacts de $\mathbb{R}^{N_x}$ et de $\mathbb{R}^{N_\xi}$ respectivement. f a donc toutes ses dérivées partielles qui sont bornées sur $\mathcal{D}_x \times \mathcal{D}_\xi$. Ce qui implique que f est k-lipschitzienne sur $\mathcal{D}_x \times \mathcal{D}_\xi$.

Pour revenir à notre égalité, nous avons donc $\exists k > 0$ tel que $\left|f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) - f(\boldsymbol{x}',\tilde{\boldsymbol{\xi}}^j)\right| \leqslant k\delta$. Sachant que le rapport de vraisemblance $L(\tilde{\boldsymbol{\xi}}^j) \leqslant M_L$ est borné pour tout $j \in \{1, \cdots, N_{IS}\}$, on a donc lorsque $\|(\boldsymbol{x},\gamma) - (\boldsymbol{x}',\gamma')\| \leqslant \delta$:

$$\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right| \leqslant \frac{\#(I_1 \cup I_2 \cup I_3)}{N_{IS}}\delta M_L(1+k) \leqslant \delta M_L(1+k) \tag{10.14}$$

Car $\frac{\#(I_1\cup I_2\cup I_3)}{N_{IS}} \leqslant 1$.

Or, par l'inégalité de Markov, nous avons :

$$\mathbb{P}\left[\sup_{\|(\boldsymbol{x},\gamma)-(\boldsymbol{x}',\gamma')\|\leqslant\delta}\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right| > \varepsilon\right] \leqslant \frac{\mathbb{E}\left\{\sup_{\|(\boldsymbol{x},\gamma)-(\boldsymbol{x}',\gamma')\|\leqslant\delta}\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right|\right\}}{\varepsilon}$$

Or d'après (10.14)

$$\sup_{\|(\boldsymbol{x},\gamma)-(\boldsymbol{x}',\gamma')\|\leqslant\delta}\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right| \leqslant \delta M_L(1+k)$$

On a donc

$$\mathbb{P}\left[\sup_{\|(\boldsymbol{x},\gamma)-(\boldsymbol{x}',\gamma')\|\leqslant\delta}\left|\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi^{N_{IS}}(\boldsymbol{x}',\gamma')\right| > \varepsilon\right] \leqslant \frac{\delta M_L(1+k)}{\varepsilon} = \eta \tag{10.15}$$

Donc $\forall\varepsilon > 0$, $\forall\eta > 0$, $\exists 1 > \delta > 0 : \delta = \frac{\varepsilon\eta}{M'_L(1+k)}$ avec $M'_L = \max\left(M_L, \frac{\varepsilon\eta}{1+k}\right)$. □

Ceci montre donc la relative compacité. Ce qui nous permet d'énoncer le théorème suivant :

Théorème 10.7. *Si $f \in \mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$ avec $\mathcal{D}_x$ et $\mathcal{D}_\xi$ deux compacts de $\mathbb{R}^{N_x}$ et de $\mathbb{R}^{N_\xi}$ respectivement.*

alors $\Psi^{N_{IS}}$ converge vers un processus aléatoire défini dans $\mathcal{C}^1(\mathcal{D}_x \times \mathcal{D}_\xi, \mathbb{R})$.

Preuve.
Pour prouver la convergence, d'après le théorème 10.4, il est suffisant de montrer qu'il y a unicité de la valeur d'adhérence et que la suite est relativement compacte.

Pour la preuve de la compacité, on prouve d'abord la tension de la suite $\left(\Psi^{N_{IS}}\right)$ au théorème 10.6. Puis par application du théorème de Prokhorov (théorème 10.5), on obtient la relative compacité.

Pour l'unicité, le lemme 4.2 prouve la convergence de ce processus $\forall\,(\boldsymbol{x},\gamma) \in \mathcal{D}_x \times \mathbb{R}$. Montrons à présent que cela suffit à prouver l'unicité de la valeur d'adhérence. Ceci se montre par l'absurde.

Supposons qu'il n'existe pas une unique valeur d'adhérence. Ceci signifie qu'il existe au moins deux valeurs d'adhérence $g_1\,(\boldsymbol{x},\gamma)$ et $g_2\,(\boldsymbol{x},\gamma)$ supposées distinctes. Ceci signifie notamment qu'il existe un $(\boldsymbol{x},\gamma) \in \mathcal{D}_x \times \mathbb{R}$ tel que $g_1\,(\boldsymbol{x},\gamma) \neq g_2\,(\boldsymbol{x},\gamma)$. Or, par le lemme 10.5, $\Psi^{N_{IS}}\,(\boldsymbol{x},\gamma)$ tend vers une valeur déterministe donnée $\Psi(\boldsymbol{x},\gamma)$ pour chaque $(\boldsymbol{x},\gamma) \in \mathcal{D}_x \times \mathbb{R}$. Il existe donc une unique valeur possible pour la convergence en chaque point. Ceci contredit l'hypothèse qu'il existe deux valeurs d'adhérence. Il ne peut donc en exister qu'une seule. □

10.4 Réutilisation des points et échantillonnage préférentiel

Dans cette partie, l'objectif est d'établir les conditions permettant de garantir que l'estimateur par échantillonnage préférentiel reste convergent lorsque des données provenant de différentes lois biaisées sont utilisées. On se place donc dans le cas où les incertitudes sont générées suivant des densités dépendantes du point autour duquel elles ont été générées. On doit donc introduire de nouvelles notations : soit $h_b(\bullet;\ \boldsymbol{x})$ la loi biaisée suivant laquelle on tire les échantillons et $h(\bullet;\ \boldsymbol{x})$ la densité réelle des incertitudes. On a donc le rapport de vraisemblance $L(\bullet;\ \boldsymbol{x})$ qui dépend également du point $\boldsymbol{x}$ autour duquel les échantillons ont été générés.

Pour se placer dans un cas similaire à celui de l'optimisation, supposons que les échantillons ont été générés autour de $d \in \mathbb{N}$ points $\left(\boldsymbol{x}^k\right)_{1\leqslant k\leqslant d}$. Soient $\left(\tilde{\boldsymbol{\xi}}^{k,i}\right)_{1\leqslant i\leqslant N(k)}$ les $N(k)$ réalisations des incertitudes générées autour du point $\boldsymbol{x}^k$ pour estimer la mesure de risque autour du point $\boldsymbol{x}^k$, suivant la densité biaisée $h_b\left(\bullet;\ \boldsymbol{x}^k\right)$. Dans le cas où l'on souhaite échantillonner simultanément autour des d différents points de design, il faut faire attention à échantillonner autour de chaque $\boldsymbol{x}^k$ de sorte que ces échantillons respectent la condition de Lindeberg donnée dans le livre [Feller, 1966, pp.262-264].

Définition 10.3 (Condition de Lindeberg)**.** Soient $\left(\boldsymbol{X}^1, \cdots, \boldsymbol{X}^k, \cdots, \boldsymbol{X}^n\right)$ des variables aléatoires indépendantes et telles que $\mathbb{E}\left(\boldsymbol{X}^k\right) = 0$ et $\mathbb{E}\left(\left(X^k\right)^2\right) = \sigma_k^2$. La condition de Lindeberg est vérifiée si :

$$\lim_{n\to\infty} \frac{1}{\sum_{i=1}^n \sigma_i^2} \sum_{i=1}^n \mathbb{E}\left[\left(X^k\right)^2 \mathbf{1}_{|X^k|>\varepsilon \sum_{i=1}^n \sigma_i^2}\right] = 0 \quad \forall \varepsilon > 0 \tag{10.16}$$

Dans notre cas, on échantillonne selon $d \in \mathbb{N}^\star$ (d fixé) lois distinctes deux à deux. Soit $N(k)$ (avec $k = \{1, \cdots, d\}$) le nombre d'échantillons générés selon chacune de ces lois. On a donc $n = \sum_{i=1}^d N(i)$ et la condition de Lindeberg devient :

$$\lim_{N_{IS}\to\infty} \frac{1}{\sum_{i=1}^d N(i)\sigma_i^2} \sum_{i=1}^d N(i)\mathbb{E}\left[\left(X^i\right)^2 \mathbf{1}_{|X^k|>\varepsilon \sum_{i=1}^d N(i)\sigma_i^2}\right] = 0 \quad \forall \varepsilon > 0 \tag{10.17}$$

Proposition 10.1. *Si les $n = \sum_{i=1}^d N(i)$ échantillons sont générés selon d lois distinctes, avec $d \in \mathbb{N}^\star$ fixé et telles que pour $k \in \{1, \cdots, d\}$ $\mathbb{E}\left(\boldsymbol{X}^k\right) = 0$ et $\mathbb{E}\left(\left(X^k\right)^2\right) = \sigma_k^2 < \infty$. Alors la condition de Lindeberg est vérifiée quelle que soit la manière dont le nombre total d'échantillons est généré.*

Preuve.

Posons $\Sigma_n = \frac{1}{\sum_{k=1}^d N(k)\sigma_k^2} \sum_{k=1}^d N(k)\mathbb{E}\left[\left(X^k\right)^2 \mathbf{1}_{|X^k|>\varepsilon \sum_{i=1}^d N(i)\sigma_i^2}\right]$.

$$\begin{aligned}
\Sigma_n &\leqslant \frac{\max_{k\in\{1,\cdots,d\}} \mathbb{E}\left[\left(X^k\right)^2 \mathbf{1}_{|X^k|>\varepsilon \sum_{i=1}^d N(i)\sigma_i^2}\right]}{\min_{k\in\{1,\cdots,d\}} \sigma_k^2} \\
&\leqslant C_1 \max_{k\in\{1,\cdots,d\}} \mathbb{E}\left[\left(X^k\right)^2 \mathbf{1}_{|X^k|>\varepsilon \sum_{i=1}^d N(i)\sigma_i^2}\right]
\end{aligned}$$

Car les σ_k sont finis par hypothèse.

Enfin $\forall k \in \{1, \cdots, d\}$, $\mathbb{E}\left[\left(X^k\right)^2 \mathbf{1}_{|X^k|>\varepsilon \sum_{i=1}^d N(i)\sigma_i^2}\right] \underset{n\to\infty}{\longrightarrow} 0$. Sachant que d est fini, on a donc $\Sigma_n \underset{n\to\infty}{\longrightarrow} 0$, quelle que soit la manière dont $n = \sum_{i=1}^d N(i)$ tend vers l'infini. Ceci prouve le résultat annoncé.

□

Cas du quantile : L'estimateur de la distribution de la sortie autour d'un point $\boldsymbol{x} \in \mathcal{D}_x$ est donné par la formule suivante :

$$\tilde{H}_{N_{IS}}(y;\ \boldsymbol{x}) = \frac{1}{\sum_{k=1}^d N(k)} \sum_{k=1}^{d} \sum_{i=1}^{N(k)} L(\tilde{\xi}^{k,i};\ \boldsymbol{x}^k)\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{k,i})\leqslant y}$$

Théorème 10.8. *Si* $\forall k \in \{1...d\}$, $\mathbb{E}[L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^k)^3] < \infty$ *et* $H(\bullet;\ \boldsymbol{x})$ *différentiable en* $q_\alpha(\boldsymbol{x}) = H^{-1}(\alpha;\ \boldsymbol{x})$ *avec* $H'(q_\alpha(\boldsymbol{x});\ \boldsymbol{x}) = h_Y(q_\alpha(\boldsymbol{x});\ \boldsymbol{x}) > 0$.
Soit $\sigma^2_{N_{IS}}(\boldsymbol{x}) = \frac{1}{N_{IS}} \sum_{k=1}^d N(k)\sigma_k^2(\boldsymbol{x})$ *et* $\sigma(\boldsymbol{x}) = \lim_{N_{IS}\to\infty} \sigma_{N_{IS}}(\boldsymbol{x}) < \infty$

$$\textit{Avec } \sigma_k^2(\boldsymbol{x}) = \frac{\tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^k)^2 \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{k,1})\leqslant q_\alpha(\boldsymbol{x})}\right] - \alpha^2}{h_Y^2(q_\alpha(\boldsymbol{x});\ \boldsymbol{x})} \textit{ et } N_{IS} = \sum_{k=1}^d N(k)$$

Alors

$$\sqrt{N_{IS}}\left(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}) - q_\alpha(\boldsymbol{x})\right) \underset{N_{IS}\to\infty}{\overset{\mathcal{L}}{\longrightarrow}} \sigma(\boldsymbol{x})\mathcal{N}(0,1)$$

Preuve.
Comme dans la démonstration du cas d'échantillons i.i.d du théorème 4.1, on peut écrire :

$$\forall y \in \mathbb{R}\ \ \tilde{\mathbb{P}}\left[\sqrt{N_{IS}}(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}) - q_\alpha(\boldsymbol{x})) \leqslant y\right] = \tilde{\mathbb{P}}\left[\alpha \leqslant \tilde{H}_{N_{IS}}\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right)\right] \quad (10.18)$$

Introduisons les notations suivantes

$$\begin{cases} s_k(y) &= \left\{\tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^k)^2\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{k,1})\leqslant q_\alpha(\boldsymbol{x})+y/\sqrt{N_{IS}}}\right] - H^2(q_\alpha(\boldsymbol{x}) + y/\sqrt{N_{IS}};\ \boldsymbol{x})\right\}^{1/2} \\ \psi_{N_{IS}}(y) &= \frac{N_{IS}}{\sqrt{\sum_{k=1}^d N(k)s_k(y)^2}}\left[\tilde{H}_{N_{IS}}\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right) - H\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right)\right] \\ a_{N_{IS}}(y) &= \frac{N_{IS}}{\sqrt{\sum_{k=1}^d N(k)s_k(y)^2}}\left[\alpha - H\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right)\right] \end{cases}$$

Avec ces notations, on a donc :

$$\begin{aligned} \tilde{\mathbb{P}}\left[\alpha \leqslant \tilde{H}_{N_{IS}}\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right)\right] &= \tilde{\mathbb{P}}\left[a_{N_{IS}}(y) \leqslant \psi_{N_{IS}}(y)\right] \\ &\quad \underbrace{-\tilde{\mathbb{P}}\left[\mathcal{N}(0,1) \geqslant a_{N_{IS}}(y)\right] + \tilde{\mathbb{P}}\left[\mathcal{N}(0,1) \geqslant a_{N_{IS}}(y)\right]}_{=0!} \end{aligned}$$

Or, en réindexant les $\tilde{\boldsymbol{\xi}}^{k,i}$ avec un unique indice $(m,i) \to j$ (par exemple $(1,1) \to 1$, $(1,2) \to 2$, $\cdots$, $(1,N(1)) \to N(1)$,$\cdots$), on peut réécrire $\psi_{N_{IS}}(y)$:

$$\psi_{N_{IS}}(y) = \frac{\sum_{j=1}^{N_{IS}} L(\tilde{\boldsymbol{\xi}}^j;\ \boldsymbol{x}^{k_j})\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j)\leqslant q_\alpha(\boldsymbol{x})+y/\sqrt{N_{IS}}} - \tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^j;\ \boldsymbol{x}^{k_j})\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j)\leqslant q_\alpha(\boldsymbol{x})+y/\sqrt{N_{IS}}}\right]}{\left(\sum_{k=1}^d N(k)\,\mathrm{Var}\left[L(\tilde{\boldsymbol{\xi}}^j;\ \boldsymbol{x}^{k_j})\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{x}}^{k_j,1})\leqslant q_\alpha(\boldsymbol{x})+y/\sqrt{N_{IS}}}\right]\right)^{1/2}}$$

$$\text{Car} \begin{cases} \tilde{H}_{N_{IS}}\left(q_\alpha(\boldsymbol{x}) + \dfrac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right) = \dfrac{1}{N_{IS}} \displaystyle\sum_{k=1}^{d} \sum_{i=1}^{N(k)} L(\tilde{\xi}^{k,i};\ \boldsymbol{x}^k)\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{k,i}) \leqslant q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}}} \\ H(q_\alpha(\boldsymbol{x}) + y/\sqrt{N_{IS}}) = \tilde{\mathbb{E}}\left[L(\tilde{\boldsymbol{\xi}}^j;\ \boldsymbol{x}^{k_j})\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) \leqslant q_\alpha(\boldsymbol{x}) + y/\sqrt{N_{IS}}}\right] \\ s_k^2(y) = \mathrm{Var}\left[L(\tilde{\boldsymbol{\xi}}^j;\ \boldsymbol{x}^{k_j})\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) \leqslant q_\alpha(\boldsymbol{x}) + y/\sqrt{N_{IS}}}\right] \end{cases}$$

En appliquant le théorème de Berry-Esseen généralisé provenant du livre [Feller, 1966, pp.544-545], on a donc l'inégalité suivante (en omettant les dépendances en $\boldsymbol{x}$ afin de simplifier l'écriture) :

$$\sup_u \left|\tilde{\mathbb{P}}\left[\psi_{N_{IS}}(y) \leqslant \boldsymbol{u}\right] - \Phi_{(0,1)}(\boldsymbol{u})\right| \leqslant C \frac{\sum_{j=1}^{N_{IS}} \rho_j}{\left(\sqrt{\sum_{j=1}^{N_{IS}} \sigma_j}\right)^3}$$

$\Phi_{(0,1)}$ correspond à la fonction de répartition de la loi normale (définie à l'annexe A) et $\tilde{\rho}_j = \tilde{\mathbb{E}}\left(\left[L(\tilde{\boldsymbol{\xi}}^j;\ \boldsymbol{x}^{k_j})\ \mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^j) \leqslant q_\alpha(\boldsymbol{x}) + y/\sqrt{N_{IS}}}\right]^3\right)$.

Or, grâce aux hypothèses données sur les rapports de vraisemblance (i.e $\forall k \in \{1...d\}$ $\mathbb{E}[L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^k)^3] < \infty$), on a donc que $\forall k \in \{1...d\}$ $\mathbb{E}[L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^k)^3] = M_k < \infty$ avec d fixé (le nombre d'échantillons biaisés différents utilisés). Soit $M_\rho = \sup_{k \in \{1...d\}} \{k\ :\ M_k\}$ et, en faisant de même pour σ_j, soit M_σ le sup des $\mathbb{E}[L(\tilde{\boldsymbol{\xi}}^{k,1}; \boldsymbol{x}^k)^2]$.

$$\begin{aligned} \Longrightarrow C \frac{\sum_{j=1}^{N_{IS}} \tilde{\rho}_j}{\left(\sqrt{\sum_{j=1}^{N_{IS}} \tilde{\sigma}_j}\right)^3} &\leqslant C \frac{N_{IS} M_\rho}{N_{IS}^{3/2} M_\sigma^3} \\ &\leqslant \frac{C'}{\sqrt{N_{IS}}} \end{aligned}$$

Avec $C' > 0$.
Donc lorsque $N_{IS} \to \infty$, $N_{IS} \to \infty$. Ainsi, en repartant de l'équation (10.11), on a donc

$$\tilde{\mathbb{P}}\left(\sqrt{N_{IS}}(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}) - q_\alpha(\boldsymbol{x})) \leqslant y\right) \underset{N_{IS}\infty}{=} \Phi_{(0,1)}\left(a_{N_{IS}}(y)\right)$$

Et pour le comportement de $a_{N_{IS}}(y)$ quand $N_{IS} \to \infty$, on peut écrire :

$$\begin{aligned} a_{N_{IS}}(y) &= \frac{N_{IS}}{\sqrt{\sum_{k=1}^d N(k) s_k(y)^2}} \left[\alpha - H\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{tot}}};\ \boldsymbol{x}\right)\right] \\ &= \frac{y}{\sqrt{1/N_{IS} \sum_{k=1}^d N(k) s_k(y)^2}} \frac{H(q_\alpha(\boldsymbol{x}); \boldsymbol{x}) - H\left(q_\alpha(\boldsymbol{x}) + \frac{y}{\sqrt{N_{IS}}};\ \boldsymbol{x}\right)}{\frac{y}{\sqrt{N_{IS}}}} \\ &\underset{N_{IS}\to\infty}{\longrightarrow} -\frac{y}{\sigma(\boldsymbol{x})} \end{aligned}$$

Ce qui nous donne en revenant à l'équation (10.11) pour N_{IS} tendant vers ∞ :

$$\begin{aligned} \tilde{\mathbb{P}}\left(\sqrt{N_{IS}}(\tilde{q}_\alpha^{N_{IS}}(\boldsymbol{x}) - q_\alpha(\boldsymbol{x})) \leqslant y\right) &= \tilde{\mathbb{P}}\left(\mathcal{N}(0,1) \geqslant -\frac{y}{\sigma(\boldsymbol{x})}\right) + O\left(\frac{1}{\sqrt{N_{IS}}}\right) \\ &= \tilde{\mathbb{P}}\left(\sigma(\boldsymbol{x})\mathcal{N}(0,1) \leqslant y\right) + O\left(\frac{1}{\sqrt{N_{IS}}}\right) \end{aligned} \tag{10.19}$$

Avec $\sigma(\boldsymbol{x}) = \lim_{N_{IS}\to\infty} \sigma_{N_{IS}}(\boldsymbol{x})$ □

Cas du superquantile : pour le superquantile, comme dans la démonstration du théorème 4.2 dans le cas i.i.d, c'est par l'estimation par échantillonnage préférentiel de la fonction Ψ introduite par Rockafellar [Rockafellar et Uryasev, 2000] que l'on prouve la convergence. Dans le cadre d'échantillons tirés selon différentes loi biaisées, on a :

$$\Psi^{N_{IS}}(\boldsymbol{x},\gamma) = \gamma + \frac{1}{(1-\alpha)\sum_{k=1}^{d} N(k)} \sum_{k=1}^{d} \sum_{j=1}^{N(k)} \left[f(\boldsymbol{x}, \tilde{\boldsymbol{\xi}}^{k,j}) - \gamma \right]^{+} L\left(\tilde{\boldsymbol{\xi}}^{k,j};\ \boldsymbol{x}^{k}\right)$$

Théorème 10.9. *Si* $\forall \boldsymbol{x} \in \mathcal{D}_x$ *et* $\forall k \in \{1,...,d\}$, $\mathbb{E}\left[\left(f(\boldsymbol{x}, \tilde{\boldsymbol{\xi}}^{k,1}) - \gamma\right)^2 L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^{k})^2\right] < \infty$
Soit $\zeta^2_{N_{IS}}(\boldsymbol{x}) = \sum_{k=1}^{d} N(k)\sigma_k^2(\boldsymbol{x})$

$$\begin{aligned} \textit{avec } \sigma_k^2(\boldsymbol{x}) \quad = \quad & \frac{1}{(1-\alpha)^2}\Bigg(\mathbb{E}\left[\mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{k,1})\geqslant\gamma}\left(f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{k,1}) - \gamma\right)^2 L(\tilde{\boldsymbol{\xi}}^{k,1};\ \boldsymbol{x}^{k})^2\right] \\ & - \left[\int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}} \mathbf{1}_{f(\boldsymbol{x},\boldsymbol{\chi})\geqslant\gamma}\left(f(\boldsymbol{x},\boldsymbol{\chi}) - \gamma\right) L(\boldsymbol{\chi};\ \boldsymbol{x}) h_b(\boldsymbol{\chi};\ \boldsymbol{x})\, d\boldsymbol{\chi}\right]^2\Bigg) \end{aligned}$$

Alors

$$\frac{N_{IS}}{\zeta_{N_{IS}}}\left(\Psi^{M,n}(\boldsymbol{x},\gamma) - \Psi(\boldsymbol{x},\gamma)\right) \xrightarrow[N_{IS}\to\infty]{\mathcal{L}} \mathcal{N}(0,1)$$

Remarque : La quantité $\frac{N_{IS}}{\zeta_{N_{IS}}}$ est bien homogène à $\frac{\sqrt{N_{IS}}}{\sigma}$ puisque $\zeta_{N_{IS}} = \sqrt{\sum_{k=1}^{d} N(k)\sigma_k^2(\boldsymbol{x})}$.

Preuve.
En réindexant les $\tilde{X}^{k,j}$ avec un unique indice $(k,j) \to i$ (par exemple $(1,1) \to 1$, $(1,2) \to 2$, $\cdots$, $(1,N(1)) \to N(1)$,$\cdots$). Posons $U_i = \frac{\mathbf{1}_{f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{i})\geqslant\gamma}\left(f(\boldsymbol{x},\tilde{\boldsymbol{\xi}}^{i})-\gamma\right)L(\tilde{\boldsymbol{\xi}}^{i};\ \boldsymbol{x}^{k_i})}{1-\alpha}$. Alors $U_1, \cdots, U_{N_{IS}}$ est une suite de variables i.i.d de variance $\sigma^2_{k_i}$. On a donc :

$$\mathbb{E}(U_i) = \frac{1}{1-\alpha}\int_{\boldsymbol{\chi}\in\mathbb{R}^{N_\xi}} \mathbf{1}_{f(\boldsymbol{x},\boldsymbol{\chi})\geqslant\gamma}\left(f(\boldsymbol{x},\boldsymbol{\chi}) - \gamma\right) L(\boldsymbol{\chi};\ \boldsymbol{x}^{k_i}) h_b(\boldsymbol{\chi};\ \boldsymbol{x})\, d\boldsymbol{\chi}$$

Et on en déduit donc que :

$$\sum_{k=1}^{d} N(k)\left(\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi(\boldsymbol{x},\gamma)\right) = \sum_{i=1}^{N_{IS}} \underbrace{(U_i - \mathbb{E}(U_i))}_{Z_i}$$

Avec la variable Z_i de variance $0 < \sigma_{k_i} < \infty$ $\forall i$. La proposition 10.1 prouve que la condition de Lindeberg est vérifiée quelle que soit la manière dont N_{IS} tend vers l'infini. Par application directe du théorème central limite sur U_i, on obtient :

$$\frac{N_{IS}}{\zeta_{N_{IS}}}\left(\Psi^{N_{IS}}(\boldsymbol{x},\gamma) - \Psi(\boldsymbol{x},\gamma)\right) \xrightarrow[N_{IS}\to\infty]{\mathcal{L}} \mathcal{N}(0,1)$$

□

Conclusions et perspectives

Cette thèse a permis de proposer des algorithmes d'optimisation multi-objectifs permettant de prendre en compte les incertitudes sur les paramètres de contrôle et environnementaux par des mesures de robustesse telles que le pire cas ou le superquantile, en particulier dans le cas où les fonctions objectif sont coûteuses à calculer.

Pour cela, nous avons mis en place un algorithme multi-objectif, appelé MOEGO NSGA-II, couplant les méthodes MOEGO pour la parcimonie et NSGA-II pour l'exploration de l'espace des paramètres de contrôle. Ce dernier permet notamment de :

1. nécessiter peu d'appels distribués aux fonctions objectif afin de converger vers une discrétisation suffisante du front de Pareto complet.
2. rajouter simultanément plusieurs points optimaux pour l'amélioration espérée multi-objectif au plan d'expériences des modèles de krigeage.
3. rajouter uniquement les individus informatifs en filtrant les points les plus proches dans l'étape d'enrichissement du plan d'expériences.
4. distribuer les appels aux fonctions objectif.

De plus, même si nous n'avons pas considéré de contraintes ou un nombre d'objectif supérieur à deux dans nos cas tests d'application de MOEGO NSGA-II, ce ne sont pas des limitations à notre approche et la méthode reste utilisable. La section 6.4.5 propose une manière de prendre en compte ces cas de figure.

Nous avons également mis en place un couplage entre l'algorithme d'optimisation multi-objectif et l'estimation de la mesure de risque en chaque point de contrôle et reposant sur la réutilisation des simulations numériques coûteuses provenant de l'évaluation de la mesure de robustesse. Cette réutilisation est applicable aussi bien pour le pire cas que pour le superquantile.

En ce qui concerne le superquantile, un estimateur parcimonieux, basé sur le krigeage et l'échantillonnage préférentiel, a été développé et validé pour quantifier les incertitudes sur l'une des fonctions objectif du problème industriel. Ainsi, les fronts de Pareto obtenus par les mesures de risque pire cas et superquantile ont pu être comparés sur un cas test analytique que nous avons construit et sur le problème industriel. En particulier, ceci a permis de mettre en évidence que le pire cas pouvait retourner des solutions trop conservatives, c'est-à-dire dont la probabilité d'occurrence est très faible, alors que le superquantile a permis de proposer des solutions innovantes (car non accessibles par le pire cas) et plus performantes (par définition) que celles retournées par le pire cas. Toutefois, l'optimisation pire cas présente l'avantage d'être moins coûteuse que celle basée sur le superquantile. Si l'on considère le temps de restitution d'une discrétisation régulière du front de Pareto complet, nous avons montré qu'il était équivalent à celui de 2000 appels à la fonction coûteuse pour le superquantile, alors qu'il est plus proche de 1000 appels lorsque

l'on utilise le pire cas comme mesure de risque pour les incertitudes. En outre, même si dans cette étude uniquement un seul des objectifs était incertain, cette méthode s'étend naturellement au cas de plusieurs critères incertains. Les mesures de robustesse utilisées pour chaque objectif peuvent d'ailleurs être différentes.

Enfin, pour le problème industriel, les simulations numériques permettant d'évaluer la température du composant critique de l'équipement au cours du temps pour l'estimation de la durée de vie peuvent être trop coûteuses pour envisager l'optimisation multi-objectif sous incertitudes. Pour cela, une méthodologie de construction de modèles de substitution spatio-temporels par réseaux de neurones récurrents a été mise en place et validée.

En ce qui concerne les perspectives, il serait d'abord intéressant de ne plus avoir à définir de familles paramétriques pour l'estimation du superquantile. En effet, c'est le choix le plus difficile à réaliser dans le cadre de l'algorithme d'entropie croisée. Une façon de faire serait d'adapter l'échantillonnage stratifié, détaillé dans la section 4.2.2.1, à l'estimation de superquantiles. Bien qu'il ressemble à la méthode de l'entropie croisée, il présente l'avantage de ne pas avoir à choisir de familles paramétriques pour la loi biaisée. De plus, comme l'échantillonnage préférentiel, il nécessite moins d'appels que des techniques de Monte-Carlo pour obtenir une précision équivalente sur l'estimation de la mesure de risque [Morio et Balesdent, 2015]. Toutefois, cela nécessite d'élargir l'échantillonnage stratifié à l'estimation de superquantile, ce qui n'est pas direct puisque ce dernier n'est pas facilement exprimable par des produits de probabilité conditionnelle.

Une autre perspective, la plus importante, concerne le développement de méthodes adaptées au cas où le coût des simulations numériques reste trop élevé. La méthode de construction de réseaux de neurones récurrents a été développée pour répondre à cela : il permet d'estimer les profils de température dans le but d'estimer la durée de vie qui est l'objectif incertain et coûteux du problème d'optimisation industriel. Néanmoins, il serait indispensable de mesurer les erreurs du réseau de neurones en prédiction afin de pouvoir l'utiliser dans le cadre de l'optimisation. En particulier, ceci permettrait d'enrichir le plan d'expériences spatio-temporel du modèle de substitution de sorte à s'assurer que ce dernier ne commette pas une erreur importante dans les zones d'intérêt de l'espace des entrées. Pour estimer l'erreur commise en prédiction par le réseau de neurones, des techniques de type *bootstrap* (détaillées à la section 3.1.2.1) peuvent être envisagées. Cependant, ces techniques sont trop coûteuses en l'état puisqu'elles consistent à construire B réseaux de neurones récurrents à partir de B sous-ensembles du plan d'expériences, générés aléatoirement par tirage avec remise.

Dans le cas où la fonction objectif incertaine est très coûteuse, une autre solution pour se rapprocher du front de Pareto optimal en moins d'appels à cette dernière serait d'introduire un couplage plus important entre l'estimation de la mesure de risque et l'algorithme multi-objectif. Au lieu d'utiliser un modèle de substitution $\hat{f}(\boldsymbol{x}) = \max_{\xi \in \mathcal{D}_\xi} [f_1(\boldsymbol{x}, \boldsymbol{\xi})]$ de la mesure de robustesse comme au chapitre 7, une manière de faire pourrait être d'utiliser un modèle de substitution par fonction objectif avec en entrée les paramètres de contrôle $\boldsymbol{x}$ et tous les paramètres incertains $\boldsymbol{\xi} = (\boldsymbol{\xi}_x, \boldsymbol{\xi}_e)$, comme proposé par Rehman [Rehman *et al.*, 2014] en mono-objectif (voir l'algorithme 5.4 en section 5.3.2.4). Ensuite, le problème suivant serait résolu sur le modèle de substitution, par exemple par NSGA-II :

$$\begin{cases} \min\limits_{x \in \mathcal{D}_x} \max\limits_{\xi \in \mathcal{D}_\xi} \left[\hat{f}_1(\boldsymbol{x}, \boldsymbol{\xi}) \right] \\ \min\limits_{x} \hat{f}_2(\boldsymbol{x}) \end{cases} \tag{10.20}$$

Pour enrichir le plan d'expériences à chaque itération, il faudrait choisir le point $(\boldsymbol{x}^\star, \boldsymbol{\xi}^\star)$ permettant d'améliorer la connaissance du front pire cas. Pour la partie incertaine, on utiliserait le fait que le pire cas se réalise en un $\boldsymbol{\xi}^\star(\boldsymbol{x})$ en chaque $\boldsymbol{x}$, c'est-à-dire $\boldsymbol{\xi}^\star(\boldsymbol{x}) = \operatorname{argmax}_{\boldsymbol{\xi}\in\mathcal{D}_\xi} \hat{f}_1(\boldsymbol{x}, \boldsymbol{\xi})$. Pour calculer le $\boldsymbol{x}^\star$, on utiliserait pour cela l'amélioration espérée multi-objectif appliquée au processus gaussien $\hat{Y}_1(\boldsymbol{x}) \sim \mathcal{N}\left(\hat{f}_1(\boldsymbol{x}, \boldsymbol{\xi}^\star(\boldsymbol{x})), \hat{\sigma}_1(\boldsymbol{x}, \boldsymbol{\xi}^\star(\boldsymbol{x}))\right)$. Toutefois, à la différence de MOEGO NSGA-II, le front de Pareto auquel on se compare pour le critère d'amélioration espérée serait celui estimé sur la surface de réponse.

Ceci explique donc une limite importante de cet algorithme : il retournerait des solutions obtenues sur le modèle de substitution. C'est-à-dire qu'il ne résoudrait pas le problème de l'équation (7.1) mais celui de l'équation (10.20). Nous ne sommes donc pas assurés que le front de Pareto retourné soit aussi Pareto-optimal sur les fonctions de référence des objectifs. En d'autres termes, ce que retournerait l'algorithme 5.4 adapté au problème multi-objectif pire cas ne serait pas forcément solution du problème initial (celui de l'équation (7.1)). Cette limite est en fait analogue à ce que montre Jones dans son étude [Jones, 2001] en optimisation mono-objectif et détaillé à la section 5.1.3. De la même manière que Mohammadi [Mohammadi *et al.*, 2015] utilise EGO pour initialiser CMA-ES, le résultat de cet algorithme pourrait toutefois être récupéré pour initialiser un algorithme plus coûteux en nombre d'appels aux fonctions objectif mais plus précis comme MOEGO NSGA-II couplé avec EGO.

Pour être adapté à une mesure de risque probabiliste ρ_ξ, du type du superquantile, il serait nécessaire d'apporter davantage de modifications à l'algorithme 5.4 que pour le pire cas. En particulier :

— l'équation (10.20) doit être remplacée par l'équation suivante :

$$\begin{cases} \min\limits_{x} \rho_\xi \left[\hat{f}_1(\boldsymbol{x}, \boldsymbol{\xi})\right] \\ \min\limits_{x} \hat{f}_2(\boldsymbol{x}) \end{cases} \tag{10.21}$$

Puisque le superquantile est résolu sur le métamodèle, il est possible de l'estimer par Monte-Carlo cru avec un nombre d'échantillon important.

⚠ Pour le pire cas, il est intuitif d'associer un $\hat{\boldsymbol{\xi}}^\star(\boldsymbol{x}) = \operatorname{argmax}_{\boldsymbol{\xi}} \left[\hat{f}_1(\boldsymbol{x}, \boldsymbol{\xi})\right]$ permettant d'associer un $\boldsymbol{\xi}^\star$ à chaque $\boldsymbol{x}$. Or, ceci n'est pas possible dans le cas probabiliste puisqu'il n'existe pas de point $\hat{\boldsymbol{\xi}}^\star$ où se *réalise* le superquantile, comme pour le pire cas. Cela rend le calcul du point $(\boldsymbol{x}^\star, \boldsymbol{\xi}^\star)$ difficile, notamment parce qu'il n'est plus évident de se ramener au calcul de l'EI multi-objectif.
Une piste de recherche pour calculer l'EI afin de choisir $\boldsymbol{x}^\star$ serait de considérer que le superquantile estimé sur le krigeage est également un processus gaussien et d'estimer la moyenne et la variance de l'estimation du superquantile dues au krigeage. Le point $\boldsymbol{\xi}^\star$ où la fonction de référence doit être évaluée serait quant à lui choisi une fois le point $\boldsymbol{x}^\star$ choisi. Par exemple, ceci peut être réalisé par des techniques de type SUR (4.30) présentées dans l'état de l'art et détaillées dans les études [Bect *et al.*, 2012] et [Chevalier *et al.*, 2014] ou par l'enrichissement proposé en section 8.2.2 et basé sur le critère IMSE discrétisé. Toutefois, ces techniques nécessitent un grand nombre d'échantillonnages sur le krigeage, ce qui peut rendre ces méthodes lourdes à exécuter. Elles ne sont donc adaptées qu'au cas où le temps d'exécution de la fonction f_1 est bien plus grand que le temps d'exécution du choix du point $(\boldsymbol{x}^\star, \boldsymbol{\xi}^\star)$.

Pour finir, l'optimisation multi-objectif par pire cas ou superquantile répond aux problèmes que se posent les ingénieurs pour la conception. Il est donc primordial d'y accorder un effort important à l'avenir.

Annexes

Annexe A

Lois de probabilité continues et densités usuelles

A.1 Quelques définitions : variable aléatoire continue, fonction de répartition, densité.

Une variable aléatoire X est une application de l'univers Ω dans $E \subset \mathbb{R}$.

$$\begin{array}{rccc} X & : & \Omega & \longrightarrow & E \subset \mathbb{R} \\ & & \omega & \longmapsto & X(\omega) \end{array} \tag{A.1}$$

Dans le cas où E est continu, la variable aléatoire est dite continue. Par commodité, on omet dans les notations d'une variable aléatoire de rappeler sa dépendance à l'évènement ω. Dans le cas d'une variable aléatoire continue, la probabilité des évènements élémentaires est toujours nulle. En particulier, $\forall x \in E$, $\mathbb{P}(X = x) = 0$. Lorsque E est continu, on s'intéresse donc à la probabilité que X se trouve dans un intervalle. Par exemple, si l'on choisit l'intervalle $I =]-\infty, x]$, nous obtenons la définition de la fonction de répartition de la variable aléatoire X, donnée à l'équation (A.2).

$$\mathbb{P}(X \in I) = \mathbb{P}(X \leqslant x) = \mathbb{P}(X < x) = H(x) \tag{A.2}$$

En particulier, cette fonction de répartition possède les propriétés suivantes :

1. H est continue.
2. Si $E = \mathbb{R}$, $\lim_{x \to -\infty} H(x) = 0$ et $\lim_{x \to \infty} H(x) = 1$. Si $E = [a, b]$ avec $(a, b) \in E^2$ et $a < b$, $H(a) = 0$ et $H(b) = 1$.
3. H est une fonction croissante.
4. $\forall (a, b) \in E^2$ avec $a < b$:

$$H(b) - H(a) = \mathbb{P}(a < X < b)$$

Maintenant, une variable aléatoire continue possède une densité h si sa fonction de répartition H est dérivable. Si tel est le cas, on a :

$$h(x) = \frac{dH}{dx}(x) \tag{A.3}$$

h est appelée densité de probabilité de la variable aléatoire X. Cette densité a les propriétés suivantes :

1. $\forall x \in E,\ h(x) > 0$.
2. $\mathbb{P}(a < X < b) = H(b) - H(a) = \int_a^b h(x)\ dx$.
3. $\int_{x \in E} h(x)\ dx = \mathbb{P}(X \in E) = 1$.

Enfin, la connaissance de la densité permet de définir l'espérance (équation (A.4)) et la variance (équation (A.5)) d'une variable aléatoire continue.

$$\mathbb{E}(X) = \int_{x \in E} th(t)\ dt \tag{A.4}$$

$$\mathrm{Var}(X) = \mathbb{E}\left((X - \mathbb{E}(X))^2\right) = \int_{x \in E} t^2 h(t)\ dt - \left(\int_{x \in E} th(t)\ dt\right)^2 \tag{A.5}$$

A.2 Exemples de variables aléatoires continues

Grâce aux définitions précédentes, on remarque qu'il suffit de connaître la densité de probabilité h pour connaître une variable aléatoire. De ce fait, voici une liste de variables aléatoires usuelles utilisées dans cette étude.

A.2.1 Loi uniforme

La loi uniforme permet de modéliser un phénomène qui ne favorise aucune zone de l'intervalle donné $E = [a, b]$ avec $a < b$. La variable aléatoire X suit une loi uniforme sur $[a, b]$, notée $\mathcal{U}([a, b])$, si elle a une densité constante et égale à :

$$h(x) = \frac{1}{b - a} \text{ pour } x \in [a, b] \tag{A.6}$$

Elle a pour espérance $\mathbb{E}(X) = \dfrac{b + a}{2}$ et pour variance $\mathrm{Var}(X) = \dfrac{(b - a)^2}{12}$.

A.2.2 Loi normale

La loi normale est très répandue en probabilité puisqu'elle est la loi limite de caractéristiques liées à un échantillon de grande taille [Saporta, 2006, p.43]. Elle est fréquemment utilisée en ingénierie, par exemple pour définir la variation du diamètre d'une pièce dans une fabrication industrielle, la répartition des erreurs de mesure autour de la valeur déterministe...

Elle est définie pour $E = \mathbb{R}$ et par sa moyenne vaut $\mathbb{E}(X) = \mu$ et son écart type vaut $\sqrt{\mathrm{Var}\,\mathbb{E}(X)} = \sigma$. Sa densité de probabilité h est donnée par l'équation (A.7).

$$h(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(x - \mu)^2}{2\sigma^2}\right) \tag{A.7}$$

Cette densité est représentée en figure A.1. Enfin, elle est notée $\mathcal{N}(\mu, \sigma)$.

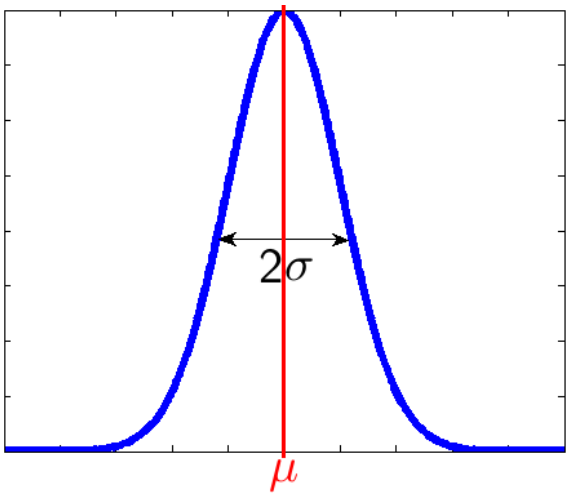


FIGURE A.1 – Représentation graphique de la densité de probabilité d'une loi normale $\mathcal{N}(\mu, \sigma)$.

A.2.3 Loi normale tronquée

Le problème de la loi normale présentée précédemment est qu'elle est définie sur $E = \mathbb{R}$. Ce qui signifie qu'il y a une probabilité non nulle pour qu'une valeur soit générée dans un intervalle $[b, \infty[$ $\forall b > 0$. Or, beaucoup de paramètres physiques sont par définition bornés et il n'est pas acceptable que l'on puisse le générer en dehors de ses bornes, ce qui signifie $E = [a, b]$ avec $a < b$. Pour cela, nous pouvons introduire la loi normale tronquée, notée ici $\mathcal{N}_{[a,b]}(\mu, \sigma)$ définie par sa moyenne μ, son écart type σ et par ses bornes $[a, b]$. Pour calculer sa densité, il suffit de renormaliser la densité de la loi normale de sorte que $\int_E h(x)\, dx = 1$. On obtient donc la densité de la loi normale tronquée $\mathcal{N}_{[a,b]}(\mu, \sigma)$ à partir de celle de la loi normale de l'équation (A.7), ici appelée $h_{\mathcal{N}(\mu,\sigma)}(x)$:

$$h(x) = \frac{h_{\mathcal{N}(\mu,\sigma)}(x)}{H_{\mathcal{N}(\mu,\sigma)}(b) - H_{\mathcal{N}(\mu,\sigma)}(a)} \tag{A.8}$$

A.2.4 Loi Bêta

Une autre loi qui peut être utilisée dans le cas où les variables aléatoires sont à support borné, par exemple $E = [0, 1]$, est la loi Bêta. Elle présente l'avantage d'avoir une forme très flexible selon la valeur des paramètres $c_1 > 0$ et $c_2 > 0$ comme l'illustre la figure A.2.

La densité de cette loi est donnée par la formule suivante :

$$\begin{aligned} h(x) &= \frac{\Gamma(c_1 + c_2)}{\Gamma(c_1)\Gamma(c_2)} x^{c_1 - 1}(1 - x)^{c_2 - 1} \\ &= \frac{1}{B(c_1, c_2)} x^{c_1 - 1}(1 - x)^{c_2 - 1} \end{aligned} \tag{A.9}$$

Γ correspond à la fonction Gamma et B correspond à la fonction Bêta. Cette dernière est une fonction de normalisation, permettant que l'intégrale de h sur le domaine E soit égale à 1.

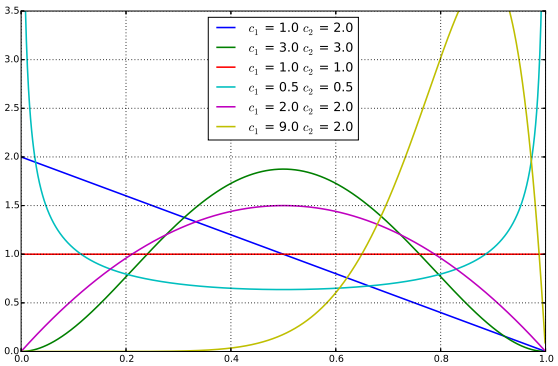


FIGURE A.2 – Représentation graphique de la densité de probabilité d'une loi bêta en fonction des paramètres α et β.

Annexe B

Estimation analytique de l'amélioration espérée bi-objectif

B.1 Résultat préliminaire

Pour les notations, la loi normale, définie à l'annexe A.2.2, a ici sa densité notée $\varphi_{(\mu,\sigma)}$ et sa fonction de répartition notée $\Phi_{(\mu,\sigma)}$. Avant de détailler le calcul de l'amélioration espérée en présence de deux objectifs, voici un résultat préliminaire.

Proposition B.1. *Soient $Y_1 \sim \mathcal{N}(\mu, \sigma)$, alors :*

$$\int_a^b y_1 \varphi_{(\mu,\sigma)}(y_1)\, dy_1 = \mu \left[\Phi_{(0,1)}\left(\frac{b-\mu}{\sigma}\right) - \Phi_{(0,1)}\left(\frac{a-\mu}{\sigma}\right)\right] - \sigma \left[\varphi_{(0,1)}\left(\frac{b-\mu}{\sigma}\right) - \varphi_{(0,1)}\left(\frac{a-\mu}{\sigma}\right)\right] \tag{B.1}$$

Preuve.

$$\begin{aligned} \int_a^b y_1 \varphi_{(0,1)}(y_1)\, dy_1 &= \int_a^b \frac{y_1}{\sigma\sqrt{2\pi}} \exp\left(-0.5\left(\frac{y_1-\mu}{\sigma}\right)^2\right) dy_1 \\ &= \mu \int_a^b \varphi_{(\mu,\sigma)}(y_1)\, dy_1 + \sigma \int_a^b \frac{y_1-\mu}{\sigma^2\sqrt{2\pi}} \exp\left(-0.5\left(\frac{y_1-\mu}{\sigma}\right)^2\right) dy_1 \end{aligned} \tag{B.2}$$

Or, par définition $\int_a^b \varphi_{(\mu,\sigma)}(y_1)\, dy_1 = \Phi_{(\mu,\sigma)}(b) - \Phi_{(\mu,\sigma)}(a)$ et

$$\frac{y_1-\mu}{\sigma^2\sqrt{2\pi}} \exp\left(-0.5\left(\frac{y_1-\mu}{\sigma}\right)^2\right) = \frac{d}{dy_1}\left[-\frac{\exp\left(\frac{y_1-\mu}{\sigma}\right)^2}{\sqrt{2\pi}}\right] = \frac{d}{dy_1}\left[-\varphi_{(0,1)}\left(\frac{y_1-\mu}{\sigma}\right)\right] \tag{B.3}$$

En repartant de l'équation (B.2) avec le résultat obtenu à l'équation (B.3), on obtient le résultat de l'équation (B.1). □

B.2 Amélioration espérée basée sur l'hypervolume

L'intégrale à estimer pour calculer l'amélioration espérée multi-objectif basée sur l'hypervolume est celle donnée par l'équation (5.16). Pour rappel, elle est de la forme :

$$\mathbb{E}\left(I_H\left(\boldsymbol{x}\right)\right) = \int_{\mathbb{R}^2} I_h(\boldsymbol{x}) \varphi_{\hat{\boldsymbol{F}}(\boldsymbol{x}),\hat{\boldsymbol{\sigma}}(\boldsymbol{x})}\left(y_1, y_2\right) dy_1 dy_2 \tag{B.4}$$

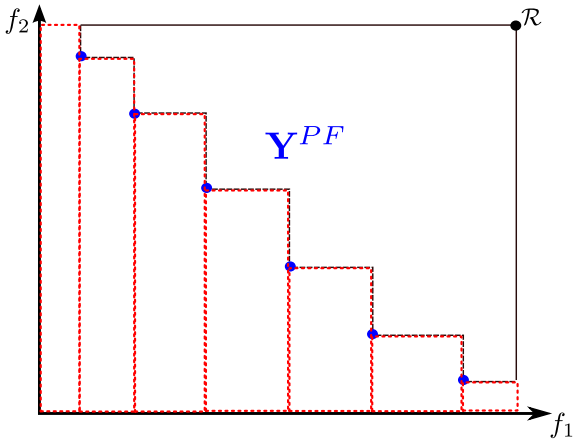


FIGURE B.1 – Illustration de la décomposition de l'intégrale de l'équation (5.16) pour le calcul de l'hypervolume.

La première hypothèse qui est faite ici est l'indépendance entre les processus de chacune des sorties. Grâce à cela, nous avons :

$$\varphi_{\hat{\boldsymbol{F}}(\boldsymbol{x}),\hat{\boldsymbol{\sigma}}(\boldsymbol{x})}(y_1, y_2) = \varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)$$

À présent, pour détailler analytiquement le calcul, nous pouvons d'abord retirer les zones où l'hypervolume ajouté $I_h(\boldsymbol{x})$ est nul (donc où $\boldsymbol{Y}(\boldsymbol{x})$ est dominé par l'un des points du front). Ensuite, pour calculer l'intégrale, il suffit de calculer ce que vaut l'hypervolume sur chacune des zones délimitées par les pointillés rouges sur la figure B.1. Cela revient à calculer l'aire d'un rectangle dont on connaît la longueur de chacun des côtés. En ordonnant les indices des sorties des points du front courant du plan d'expériences selon le premier objectif, $y_1^{PF_1} \leqslant \cdots \leqslant y_1^{PF_{N_{PF}}}$, et en omettant les PF dans leur exposant, on obtient donc :

$$\begin{aligned}
\mathbb{E}\left(I_H(\boldsymbol{x})\right) &= \int_{\mathbb{R}^2} I_h(\boldsymbol{x})\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)dy_1dy_2 \\
&= \int_{-\infty}^{y_1^1}\int_{-\infty}^{\infty}(y_1^1 - y_1)(R_2 - y_2)\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)\ dy_1dy_2 \\
&\quad + \sum_{i=1}^{N_{PF}-1}\int_{-\infty}^{y_1^i}\int_{-\infty}^{y_2^i}(y_1^{i+1} - y_1)(y_2^i - y_2)\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)\ dy_1dy_2 \\
&\quad + \int_{-\infty}^{y_1^{N_{PF}}}\int_{-\infty}^{y_2^{N_{PF}}}(R_1 - y_1)(y_2^{N_{PF}} - y_2)\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)\ dy_1dy_2
\end{aligned} \tag{B.5}$$

En utilisant le théorème de Fubini et le résultat de l'équation (B.1), nous obtenons :

$$\begin{aligned}
\mathbb{E}\left(I_H\left(\boldsymbol{x}\right)\right) &= \left\{\left(R_2-\hat{f}_2(\boldsymbol{x})\right)\Phi_{0,1}\left(\frac{R_2-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)-\hat{\sigma}_2(\boldsymbol{x})\varphi_{0,1}\left(\frac{R_2-\hat{f}_2(\boldsymbol{x}))}{\hat{\sigma}_2(\boldsymbol{x})}\right)\right\} \\
&\times\left\{\left(y_1^1-\hat{f}_1(\boldsymbol{x})\right)\Phi_{0,1}\left(\frac{y_1^1-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\hat{\sigma}_1(\boldsymbol{x})\varphi_{0,1}\left(\frac{y_1^1-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right\} \\
&\quad\left\{\left(y_2^i-\hat{f}_2(\boldsymbol{x})\right)\Phi_{0,1}\left(\frac{y_2^i-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)-\hat{\sigma}_2(\boldsymbol{x})\varphi_{0,1}\left(\frac{y_2^i-\hat{f}_2(\boldsymbol{x}))}{\hat{\sigma}_2(\boldsymbol{x})}\right)\right\} \\
&+\sum_{i=1}^{N_{PF}-1}\times\left\{\begin{array}{l}\left(y_1^{i+1}-\hat{f}_1(\boldsymbol{x})\right)\left[\Phi_{0,1}\left(\frac{y_1^{i+1}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\Phi_{0,1}\left(\frac{y_1^{i}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right] \\ -\hat{\sigma}_1(\boldsymbol{x})\left[\varphi_{0,1}\left(\frac{y_1^{i+1}-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\varphi_{0,1}\left(\frac{y_1^{i}-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\end{array}\right\} \\
&+\left\{\left(y_2^{N_{PF}}-\hat{f}_2(\boldsymbol{x})\right)\Phi_{0,1}\left(\frac{y_2^{N_{PF}}-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)-\hat{\sigma}_2(\boldsymbol{x})\varphi_{0,1}\left(\frac{y_2^{N_{PF}}-\hat{f}_2(\boldsymbol{x}))}{\hat{\sigma}_2(\boldsymbol{x})}\right)\right\} \\
&\times\left\{\begin{array}{l}\left(R_1-\hat{f}_1(\boldsymbol{x})\right)\left[\Phi_{0,1}\left(\frac{R_1-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\Phi_{0,1}\left(\frac{y_1^{N_{PF}}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right] \\ -\hat{\sigma}_1(\boldsymbol{x})\left[\varphi_{0,1}\left(\frac{R_1-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\varphi_{0,1}\left(\frac{y_1^{N_{PF}}-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\end{array}\right\}
\end{aligned} \tag{B.6}$$

où les points $\left(\boldsymbol{y}^1,\cdots,\boldsymbol{y}^{N_{PF}}\right)$ sont les points du DOE courant Pareto-optimaux et ordonnés par ordre croissant selon le premier objectif.

B.3 Amélioration espérée basée sur la distance euclidienne

Pour ce critère là, repartons de l'équation (5.14) :

$$\mathbb{E}\left[I_E(x)\right]=\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x})\prec\boldsymbol{Y}^{PF}\right)d\left(\bar{\boldsymbol{y}}(\boldsymbol{x}),\boldsymbol{Y}^{PF}\right) \tag{B.7}$$

On remarque qu'il y a donc deux composantes à estimer : la probabilité que le point $\boldsymbol{x}$ soit Pareto-optimal et la moyenne de la distance entre le front et le processus de sortie.

B.3.1 Estimation de la probabilité de dominance

La probabilité à estimer est la suivante :

$$\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x})\prec\boldsymbol{Y}^{PF}\right)=\int_{\mathbb{R}^2}\mathbf{1}_{\boldsymbol{Y}(\boldsymbol{x})\prec\boldsymbol{Y}^{PF}}\varphi_{\hat{\boldsymbol{F}}(\boldsymbol{x}),\hat{\boldsymbol{\sigma}}(\boldsymbol{x})}\left(y_1,y_2\right)dy_1dy_2 \tag{B.8}$$

Comme pour l'hypervolume, en considérant que les processus gaussiens des deux sorties sont indépendants et en décomposant $\mathbb{R}^2$ en sous-domaines représentés en rouge sur la figure B.2, nous avons :

$$\begin{aligned}
\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x})\prec\boldsymbol{Y}^{PF}\right) &= \int_{-\infty}^{y_1^1}\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\;dy_1\int_{-\infty}^{\infty}\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)\;dy_2 \\
&+\sum_{i=1}^{N_{PF}-1}\int_{-\infty}^{y_1^i}\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\;dy_1\int_{-\infty}^{y_2^i}\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)\;dy_2 \\
&+\int_{-\infty}^{y_1^{N_{PF}}}\varphi_{\hat{f}_1(\boldsymbol{x}),\hat{\sigma}_1(\boldsymbol{x})}(y_1)\;dy_1\int_{-\infty}^{y_2^{N_{PF}}}\varphi_{\hat{f}_2(\boldsymbol{x}),\hat{\sigma}_2(\boldsymbol{x})}(y_2)\;dy_2
\end{aligned} \tag{B.9}$$

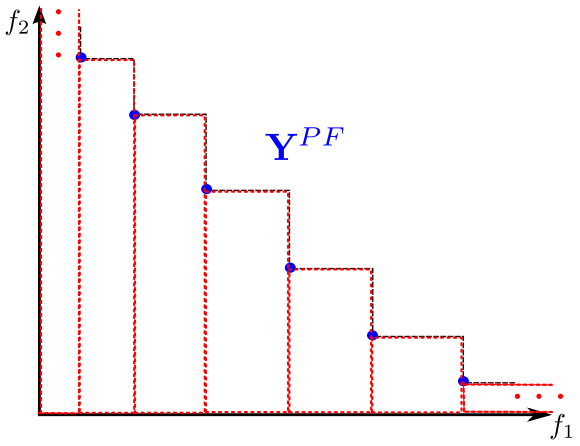


FIGURE B.2 – Illustration de la décomposition de l'intégrale pour le calcul de la probabilité de l'équation (B.8) et de la distance de l'équation (B.11).

Par définition de la densité φ, nous obtenons donc directement pour cette probabilité :

$$\begin{aligned}\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}\right) &= \Phi_{0,1}\left(\frac{y_1^1 - \hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right) + \\ &+ \sum_{i=1}^{N_{PF}-1} \Phi_{0,1}\left(\frac{y_2^i - \hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)\left[\Phi_{0,1}\left(\frac{y_1^{i+1} - \hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right) - \Phi_{0,1}\left(\frac{y_1^i - \hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right] \\ &+ \Phi_{0,1}\left(\frac{y_2^{N_{PF}} - \hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)\left[1 - \Phi_{0,1}\left(\frac{y_1^i - \hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\end{aligned} \tag{B.10}$$

B.3.2 Estimation de la distance

Pour la distance, on a :

$$d\left(\bar{\boldsymbol{y}}(\boldsymbol{x}), \boldsymbol{Y}^{PF}\right) = \sqrt{\left(\bar{y}_1 - y_1^\star\right)^2 + \left(\bar{y}_2 - y_2^\star\right)^2} \tag{B.11}$$

$\bar{\boldsymbol{y}} = (\bar{y}_1, \bar{y}_2)$ est la position moyenne du processus dominant $\boldsymbol{Y}(\boldsymbol{x})$. $\boldsymbol{y}^\star$ est le point de $\boldsymbol{Y}^{PF}$ le plus proche de $\bar{\boldsymbol{y}}$ par la distance euclidienne. $\bar{\boldsymbol{y}}$ se calcule par l'intégrale suivante :

$$\begin{cases} \bar{y}_1 = \dfrac{1}{\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}\right)} \displaystyle\int_{\mathbb{R}^2} y_1 \mathbf{1}_{\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}} \varphi_{\hat{\boldsymbol{F}}(\boldsymbol{x}), \hat{\boldsymbol{\sigma}}(\boldsymbol{x})}(y_1, y_2)\, dy_1 dy_2 \\ \bar{y}_2 = \dfrac{1}{\mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}\right)} \displaystyle\int_{\mathbb{R}^2} y_2 \mathbf{1}_{\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}} \varphi_{\hat{\boldsymbol{F}}(\boldsymbol{x}), \hat{\boldsymbol{\sigma}}(\boldsymbol{x})}(y_1, y_2)\, dy_1 dy_2 \end{cases} \tag{B.12}$$

Pour simplifier les écritures, posons $P = \mathbb{P}\left(\boldsymbol{Y}(\boldsymbol{x}) \prec \boldsymbol{Y}^{PF}\right)$. Afin de calculer les intégrales de l'équation (B.12), nous utilisons l'indépendance entre les processus gaussiens permettant de calculer la sortie et la décomposition de $\mathbb{R}^2$ en sous-domaines représentés en rouge sur la figure B.2. De plus, par le théorème de Fubini et le résultat préliminaire

de l'équation (B.1), nous avons :

$$\begin{aligned}
\bar{y}_1 P &= \hat{f}_1(\boldsymbol{x})\,\Phi_{0,1}\left(\frac{y_1^1-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\hat{\sigma}_1(\boldsymbol{x})\,\varphi_{0,1}\left(\frac{y_1^1-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\\
&+\sum_{i=1}^{N_{PF}-1}\Phi_{0,1}\left(\frac{y_2^i-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)\times\left\{\begin{array}{l}\hat{f}_1(\boldsymbol{x})\left[\Phi_{0,1}\left(\frac{y_1^{i+1}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\Phi_{0,1}\left(\frac{y_1^{i}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\\ -\hat{\sigma}_1(\boldsymbol{x})\left[\varphi_{0,1}\left(\frac{y_1^{i+1}-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\varphi_{0,1}\left(\frac{y_1^{i}-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\end{array}\right\}\\
&+\Phi_{0,1}\left(\frac{y_2^{N_{PF}}-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)\left\{\hat{f}_1\left[1-\Phi_{0,1}\left(\frac{y_1^{i}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]-\hat{\sigma}_1(\boldsymbol{x})\varphi_{0,1}\left(\frac{y_1^{N_{PF}}-\hat{f}_1(\boldsymbol{x}))}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right\}
\end{aligned}\tag{B.13}$$

$$\begin{aligned}
\bar{y}_2 P &= \hat{f}_2(\boldsymbol{x})\,\Phi_{0,1}\left(\frac{y_1^1-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\\
&+\sum_{i=1}^{N_{PF}-1}\begin{array}{l}\left[\hat{f}_2(\boldsymbol{x})\,\Phi_{0,1}\left(\frac{y_2^{i}-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)-\hat{\sigma}_2(\boldsymbol{x})\,\varphi_{0,1}\left(\frac{y_2^{i}-\hat{f}_2(\boldsymbol{x}))}{\hat{\sigma}_2(\boldsymbol{x})}\right)\right]\\ \left[\Phi_{0,1}\left(\frac{y_1^{i+1}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)-\Phi_{0,1}\left(\frac{y_1^{i}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\end{array}\\
&+\left[1-\Phi_{0,1}\left(\frac{y_1^{N_{PF}}-\hat{f}_1(\boldsymbol{x})}{\hat{\sigma}_1(\boldsymbol{x})}\right)\right]\left[\hat{f}_2(\boldsymbol{x})\Phi_{0,1}\left(\frac{y_2^{N_{PF}}-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)-\hat{\sigma}_2(\boldsymbol{x})\varphi_{0,1}\left(\frac{y_2^{N_{PF}}-\hat{f}_2(\boldsymbol{x})}{\hat{\sigma}_2(\boldsymbol{x})}\right)\right]
\end{aligned}\tag{B.14}$$

Bibliographie

[Akhtar et Shoemaker, 2015] Akhtar, T. et Shoemaker, C. A. (2015). Multi objective optimization of computationally expensive multi-modal functions with rbf surrogates and multi-rule selection. *Journal of Global Optimization*, 64(1):17–32.

[Bachoc, 2013] Bachoc, F. (2013). Cross Validation and Maximum Likelihood estimation of hyper-parameters of Gaussian processes with model misspecification. *Computational Statistics and Data Analysis*, 66:55–69.

[Balesdent *et al.*, 2013] Balesdent, M., Morio, J. et Marzat, J. (2013). Kriging-based adaptive importance sampling algorithms for rare event estimation. *Structural Safety*, 44(0):1 – 10.

[Barron, 1993] Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *Information Theory, IEEE Transactions on*, 39(3):930–945.

[Basak *et al.*, 2007] Basak, D., Pal, S. et Patranabis, D. C. (2007). Support vector regression. *Neural Information Processing-Letters and Reviews*, 11(10):203–224.

[Battiti et Masulli, 1990] Battiti, R. et Masulli, F. (1990). *International Neural Network Conference : July 9–13, 1990 Palais Des Congres — Paris — France*, chapitre BFGS Optimization for Faster and Automated Supervised Learning, pages 757–760. Springer Netherlands, Dordrecht.

[Bect *et al.*, 2012] Bect, J., Ginsbourger, D., Li, L., Picheny, V. et Vazquez, E. (2012). Sequential design of computer experiments for the estimation of a probability of failure. *Statistics and Computing*, 22(3):773–793.

[Berry, 1941] Berry, A. C. (1941). The accuracy of the gaussian approximation to the sum of independent variates. *Transactions of the american mathematical society*, 49(1): 122–136.

[Bhattacharya, 1970] Bhattacharya, R. (1970). Rates of weak convergence for the multidimensional central limit theorem. *Theory of Probability & Its Applications*, 15(1):68–86.

[Bichon, 2010] Bichon, B. J. (2010). *Efficient surrogate modeling for reliability analysis and design*. Thèse de doctorat, Vanderbilt University.

[Billingsley, 2008] Billingsley, P. (2008). *Convergence of probability measures*. John Wiley & Sons.

[Binois, 2015] Binois, M. (2015). *Uncertainty quantification on pareto fronts and high-dimensional strategies in bayesian optimization, with applications in multi-objective automotive design*. Thèse de doctorat, Saint-Etienne, EMSE.

[Binois et Picheny, 2015] Binois, M. et Picheny, V. (2015). Gpareto : Gaussian processes for pareto front estimation and optimization. *URL http ://CRAN. R-project. org/package= GPareto. R package version*, 1(1).

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Blatman et Sudret, 2011] Blatman, G. et Sudret, B. (2011). Adaptive sparse polynomial chaos expansion based on least angle regression. *Journal of Computational Physics*, 230(6):2345 – 2367.

[Borgonovo, 2007] Borgonovo, E. (2007). A new uncertainty importance measure. *Reliability Engineering & System Safety*, 92(6):771 – 784.

[Bowman, 1984] Bowman, A. W. (1984). An alternative method of cross-validation for the smoothing of density estimates. *Biometrika*, 71(2):353–360.

[Branke *et al.*, 2013] Branke, J., Avigad, G. et Moshaiov, A. (2013). Multi-objective worst case optimization by means of evolutionary algorithms.

[Branke et Rosenbusch, 2008] Branke, J. et Rosenbusch, J. (2008). *Parallel Problem Solving from Nature – PPSN X : 10th International Conference, Dortmund, Germany, September 13-17, 2008. Proceedings*, chapitre New Approaches to Coevolutionary Worst-Case Optimization, pages 144–153. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Buhmann, 2003] Buhmann, M. D. (2003). *Radial basis functions : theory and implementations*. Cambridge monographs on applied and computational mathematics. Cambridge University Press, Cambridge, New York. Autre tirage : 2006.

[Buttazzo et Frediani, 2009] Buttazzo, G. et Frediani, A. (2009). *Variational Analysis and Aerospace Engineering*. Springer Optimization and Its Applications. Springer New York.

[Caron *et al.*, 2014] Caron, V., Guyader, A., Munoz Zuniga, M. et Tuffin, B. (2014). Some recent results in rare event estimation. *ESAIM Proceedings*, 44:239–259. Journées MAS 2012.

[Chastaing *et al.*, 2012] Chastaing, G., Gamboa, F. et Prieur, C. (2012). Generalized Hoeffding-Sobol Decomposition for Dependent Variables - Application to Sensitivity Analysis. *Electronic Journal of Statistics*, 6:2420–2448.

[Chevalier *et al.*, 2014] Chevalier, C., Bect, J., Ginsbourger, D., Vazquez, E., Picheny, V. et Richet, Y. (2014). Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. *Technometrics*, 56(4):455–465.

[Chib et Greenberg, 1995] Chib, S. et Greenberg, E. (1995). Understanding the metropolis-hastings algorithm.

[Coello *et al.*, 2013] Coello, C., Van Veldhuizen, D. et Lamont, G. (2013). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic Algorithms and Evolutionary Computation. Springer US.

[Coello Coello et Lechuga, 2002] Coello Coello, C. A. et Lechuga, M. S. (2002). Mopso : A proposal for multiple objective particle swarm optimization. *In Proceedings of the Evolutionary Computation on 2002. CEC '02. Proceedings of the 2002 Congress - Volume 02*, CEC '02, pages 1051–1056, Washington, DC, USA. IEEE Computer Society.

[Couckuyt *et al.*, 2014] Couckuyt, I., Deschrijver, D. et Dhaene, T. (2014). Fast calculation of multiobjective probability of improvement and expected improvement criteria for pareto optimization. *Journal of Global Optimization*, 60(3):575–594.

[Cybenko, 1989] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314.

[Da et Xiurun, 2005] Da, Y. et Xiurun, G. (2005). An improved pso-based ann with simulated annealing technique. *Neurocomputing*, 63:527–533.

[Da Veiga *et al.*, 2009] Da Veiga, S., Wahl, F. et Gamboa, F. (2009). Local polynomial estimation for sensitivity analysis on models with correlated inputs. *Technometrics*, 51(4):452–463.

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

[David et Nagaraja, 2004] David, H. et Nagaraja, H. (2004). *Order Statistics*. Wiley Series in Probability and Statistics. Wiley.

[Deb, 2000] Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2–4):311 – 338.

[Deb, 2011] Deb, K. (2011). Multi-objective optimisation using evolutionary algorithms : An introduction. *In Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, pages 3–34. Springer.

[Deb et Jain, 2014] Deb, K. et Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i : Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601.

[Deb *et al.*, 2002] Deb, K., Pratap, A., Agarwal, S. et Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm : Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

[Dinart, 2014] Dinart, D. (2014). Plans d'expériences spatio-temporels. Mémoire de D.E.A., Université Paul Sabatier.

[Dreyfus *et al.*, 2002] Dreyfus, G., Martinez, J., Samuelides, M., Gordon, M. B., Badran, F., Thiria, S. et Herault, L. (2002). *Réseaux de neurones - Méthodologie et applications*. Eyrolles.

[Dubourg *et al.*, 2011] Dubourg, V., Deheeger, F. et Sudret, B. (2011). Metamodel-based importance sampling for the simulation of rare events. *Applications of Statistics and Probability in Civil Engineering*, 26:192.

[Echard *et al.*, 2013] Echard, B., Gayton, N., Lemaire, M. et Relun, N. (2013). A combined importance sampling and kriging reliability method for small failure probabilities with time-demanding numerical models. *Reliability Engineering and System Safety*, 111(Complete):232–240.

[Efron, 1979] Efron, B. (1979). Bootstrap methods : Another look at the jackknife. *Ann. Statist.*, 7(1):1–26.

[Efron, 1983] Efron, B. (1983). Estimating the error rate of a prediction rule : improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331.

[Efron *et al.*, 2004] Efron, B., Hastie, T., Johnstone, I. et Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, 32:407–499.

[Elman, 1990] Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2): 179–211.

[Emmerich *et al.*, 2011] Emmerich, M. T. M., Deutz, A. H. et Klinkenberg, J. W. (2011). Hypervolume-based expected improvement : Monotonicity properties and exact computation. *In Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 2147–2154.

[Féliot *et al.*, 2016] Féliot, P., Bect, J. et Vazquez, E. (2016). A Bayesian approach to constrained single- and multi-objective optimization. working paper or preprint.

[Feller, 1966] Feller, W. (1966). *An introduction to probability theory and its applications*. Numéro vol. 2 de Wiley mathematical statistics series. Wiley.

[Finkel et Kelley, 2004] Finkel, D. E. et Kelley, C. T. (2004). Convergence Analysis of the DIRECT Algorithm. Rapport technique CRSC-TR04-28, Center for Research in Scientific Computation.

[Fletcher, 1987] Fletcher, R. (1987). *Practical Methods of Optimization ; (2Nd Ed.)*. Wiley-Interscience, New York, NY, USA.

[Forrester et Keane, 2009] Forrester, A. I. et Keane, A. J. (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79.

[Fort *et al.*, 2015] Fort, J.-C., Klein, T. et Rachdi, N. (2015). New sensitivity analysis subordinated to a contrast. *Communications in Statistics - Theory and Methods*.

[Franco, 2008] Franco, J. (2008). *Exploratory Designs for Computer Experiments of Complex Physical Systems Simulation*. Theses, Ecole Nationale Supérieure des Mines de Saint-Etienne.

[Fruth *et al.*, 2015] Fruth, J., Roustant, O. et Kuhnt, S. (2015). Sequential designs for sensitivity analysis of functional inputs in computer experiments. *Reliability Engineering and System Safety*, 134:Pages 260–267.

[Funahashi et Nakamura, 1993] Funahashi, K. et Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6(6):801 – 806.

[Gamboa *et al.*, 2014] Gamboa, F., Janon, A., Klein, T. et Lagnoux, A. (2014). Sensitivity analysis for multidimensional and functional outputs. *Electron. J. Statist.*, 8(1):575–603.

[Gamboa *et al.*, 2015] Gamboa, F., Janon, A., Klein, T., Lagnoux, A. et Prieur, C. (2015). Statistical inference for sobol pick-freeze monte carlo method. *Statistics*, pages 1–22.

[Ginsbourger *et al.*, 2010] Ginsbourger, D., Riche, R. et Carraro, L. (2010). *Computational Intelligence in Expensive Optimization Problems*, chapitre Kriging Is Well-Suited to Parallelize Optimization, pages 131–162. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Glynn, 1996] Glynn, P. W. (1996). Importance sampling for monte carlo estimation of quantiles. Rapport technique, Publishing House of Saint Petersburg University.

[Gong *et al.*, 2015] Gong, W., Duan, Q., Li, J., Wang, C., Di, Z., Ye, A., Miao, C. et Dai, Y. (2015). Multiobjective adaptive surrogate modeling-based optimization for parameter estimation of large, complex geophysical models. *Water Resources Research*.

[Grimmett et Stirzaker, 2001] Grimmett, G. et Stirzaker, D. (2001). *Probability and Random Processes*. Probability and Random Processes. OUP Oxford.

[Guerra *et al.*, 2015] Guerra, J., Klotz, P., Laurent, B. et Gamboa, F. (2015). Transient phenomena prediction using recurrent neural networks. *In 2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

[Guyader *et al.*, 2011] Guyader, A., Hengartner, N. et Matzner-Løber, E. (2011). Simulation and estimation of extreme quantiles and extreme probabilities. *Applied Mathematics & Optimization*, 64(2):171–196.

[Hagan et Menhaj, 1994] Hagan, M. T. et Menhaj, M. B. (1994). Training feedforward networks with the marquardt algorithm. *Trans. Neur. Netw.*, 5(6):989–993.

[Haldar et Mahadevan, 2000] Haldar, A. et Mahadevan, S. (2000). *Probability, reliability, and statistical methods in engineering design*. John Wiley.

[Hamdaoui *et al.*, 2014] Hamdaoui, M., Oujebbour, F.-Z., Habbal, A., Breitkopf, P. et Villon, P. (2014). Kriging surrogates for evolutionary multi-objective optimization of cpu intensive sheet metal forming applications. *International Journal of Material Forming*, 8(3):469–480.

[Han *et al.*, 2013] Han, Z.-H., Görtz, S. et Zimmermann, R. (2013). Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function. *Aerospace Science and Technology*, 25(1):177–189.

[Hansen, 2006] Hansen, N. (2006). *Towards a New Evolutionary Computation : Advances in the Estimation of Distribution Algorithms*, chapitre The CMA Evolution Strategy : A Comparing Review, pages 75–102. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Hastie *et al.*, 2009] Hastie, T. J., Tibshirani, R. J. et Friedman, J. H. (2009). *The elements of statistical learning : data mining, inference, and prediction*. Springer series in statistics. Springer, New York. Autres impressions : 2011 (corr.), 2013 (7e corr.).

[Hoeffding, 1948] Hoeffding, W. (1948). A Class of Statistics with Asymptotically Normal Distribution. *The Annals of Mathematical Statistics*, 19(3):293–325.

[Hong *et al.*, 2014] Hong, L. J., Hu, Z. et Liu, G. (2014). Monte carlo methods for value-at-risk and conditional value-at-risk : A review. *ACM Trans. Model. Comput. Simul.*, 24(4):22 :1–22 :37.

[Hopfield, 1982] Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558.

[Hornik, 1991] Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Netw.*, 4(2):251–257.

[Hupkens *et al.*, 2014] Hupkens, I., Emmerich, M. et Deutz, A. (2014). Faster computation of expected hypervolume improvement. *ArXiv e-prints, LIACS Technical Report 2004-03*.

[Hussein et Deb, 2016] Hussein, R. et Deb, K. (2016). A generative kriging surrogate model for constrained and unconstrained multi-objective optimization.

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

[Igel *et al.*, 2005] Igel, C., Wiegand, S. et Friedrichs, F. (2005). *Trends and Applications in Constructive Approximation*, chapitre Evolutionary Optimization of Neural Systems : The Use of Strategy Adaptation, pages 103–123. Birkhäuser Basel, Basel.

[Iooss et Lemaître, 2015] Iooss, B. et Lemaître, P. (2015). A review on global sensitivity analysis methods. *In* Meloni, C. et Dellino, G., éditeurs : *Uncertainty management in Simulation-Optimization of Complex Systems : Algorithms and Applications*. Springer.

[Ishibuchi *et al.*, 2008] Ishibuchi, H., Tsukamoto, N. et Nojima, Y. (2008). Evolutionary many-objective optimization : A short review. *In Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 2419–2426.

[Jaeger, 2002] Jaeger, H. (2002). *Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the "echo State Network" Approach*. GMD-Report. GMD-Forschungszentrum Informationstechnik.

[Janusevskis et Le Riche, 2013] Janusevskis, J. et Le Riche, R. (2013). Simultaneous kriging-based estimation and optimization of mean response. *Journal of Global Optimization*, 55(2):313–336.

[Johnson, 2011] Johnson, S. G. (2011). *The NLopt nonlinear-optimization package*.

[Jones, 2001] Jones, D. R. (2001). A taxonomy of global optimization methods based on response surfaces. *J. of Global Optimization*, 21(4):345–383.

[Jones *et al.*, 1993] Jones, D. R., Perttunen, C. D. et Stuckman, B. E. (1993). Lipschitzian optimization without the lipschitz constant. *J. Optim. Theory Appl.*, 79(1):157–181.

[Jones *et al.*, 1998] Jones, D. R., Schonlau, M. et Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4):455–492.

[Jones *et al.*, 01] Jones, E., Oliphant, T., Peterson, P. *et al.* (2001–). SciPy : Open source scientific tools for Python. [Online ; accessed 2016-05-17].

[Jordan, 1990] Jordan, M. I. (1990). Artificial neural networks. chapitre Attractor Dynamics and Parallelism in a Connectionist Sequential Machine, pages 112–127. IEEE Press, Piscataway, NJ, USA.

[Joyce, 2011] Joyce, J. M. (2011). *International Encyclopedia of Statistical Science*, chapitre Kullback-Leibler Divergence, pages 720–722. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Julier et Uhlmann, 1997] Julier, S. J. et Uhlmann, J. K. (1997). A new extension of the kalman filter to nonlinear systems. pages 182–193.

[Kennedy et Eberhart, 1995] Kennedy, J. et Eberhart, R. (1995). Particle swarm optimization.

[Kersaudy *et al.*, 2015] Kersaudy, P., Sudret, B., VARSIER, N., Picon, O. et Wiart, J. (2015). A new surrogate modeling technique combining Kriging and polynomial chaos expansions – Application to uncertainty analysis in computational dosimetry. *Journal of Computational Physics*, 286:130–117.

[Kleijnen, 2007] Kleijnen, J. P. C. (2007). *Design and Analysis of Simulation Experiments*. Springer Publishing Company, Incorporated, 1st édition.

[Knowles, 2006] Knowles, J. (2006). Parego : A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *Evolutionary Computation, IEEE Transactions on*, 10(1):50–66.

[Kocis et Whiten, 1997] Kocis, L. et Whiten, W. J. (1997). Computational investigations of low-discrepancy sequences. *ACM Trans. Math. Softw.*, 23(2):266–294.

[Krige, 1951] Krige, D. G. (1951). *A Statistical Approach to Some Mine Valuation and Allied Problems on the Witwatersrand.*

[Krohling *et al.*, 2004] Krohling, R. A., Hoffmann, F. et Coelho, L. S. (2004). Co-evolutionary particle swarm optimization for min-max problems using gaussian distribution. *In Evolutionary Computation, 2004. CEC2004. Congress on*, volume 1, pages 959–964 Vol.1.

[Kucherenko *et al.*, 2009] Kucherenko, S., Rodriguez-Fernandez, M., Pantelides, C. C. et Shah, N. (2009). Monte carlo evaluation of derivative-based global sensitivity measures. *Rel. Eng. & Sys. Safety*, 94(7):1135–1148.

[Kunita, 1997] Kunita, H. (1997). *Stochastic Flows and Stochastic Differential Equations.* Cambridge Studies in Advanced Mathematics. Cambridge University Press.

[Lamboni *et al.*, 2013] Lamboni, M., Iooss, B., Popelin, A.-L. et Gamboa, F. (2013). Derivative-based global sensitivity measures : General links with sobol' indices and numerical tests. *Mathematics and Computers in Simulation*, 87(0):45 – 54.

[Lecué *et al.*, 2012] Lecué, G., Mitchell, C. *et al.* (2012). Oracle inequalities for cross-validation type procedures. *Electronic Journal of Statistics*, 6:1803–1837.

[Lemaitre et Chaboche, 2004] Lemaitre, J. et Chaboche, J. (2004). *Mécanique des matériaux solides.* Sciences SUP. Sciences de l'ingénieur. Dunod.

[Liao *et al.*, 2015] Liao, S.-H., Hsieh, J.-G., Chang, J.-Y. et Lin, C.-T. (2015). Training neural networks via simplified hybrid algorithm mixing nelder–mead and particle swarm optimization methods. *Soft Computing*, 19(3):679–689.

[Ljung, 1999] Ljung, L. (1999). *System identification - Theory for the User.* Prentice-Hall.

[Loshchilov, 2013] Loshchilov, I. (2013). *Surrogate-Assisted Evolutionary Algorithms.* Thèse de doctorat, PhD Thesis. Université Paris Sud-Paris XI.

[Loshchilov *et al.*, 2013] Loshchilov, I., Schoenauer, M. et Sebag, M. (2013). Intensive surrogate model exploitation in self-adaptive surrogate-assisted cma-es (saacm-es). *In Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, pages 439–446, New York, NY, USA. ACM.

[Lozzo *et al.*, 2013] Lozzo, M. D., Klotz, P. et Laurent, B. (2013). Multilayer perceptron for the learning of spatio-temporal dynamics—application in thermal engineering. *Engineering Applications of Artificial Intelligence*, 26(10):2270 – 2286.

[Lung et Dumitrescu, 2011] Lung, R. I. et Dumitrescu, D. (2011). A new evolutionary approach to minimax problems. *In 2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 1902–1905.

[Mallipeddi et Lee, 2015] Mallipeddi, R. et Lee, M. (2015). An evolving surrogate model-based differential evolution algorithm. *Applied Soft Computing*, 34:770 – 787.

[Marler et Arora, 2004] Marler, R. et Arora, J. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395.

[Marrel *et al.*, 2008] Marrel, A., Iooss, B., Dorpe, F. V. et Volkova, E. (2008). An efficient methodology for modeling complex computer codes with gaussian processes. *Computational Statistics and Data Analysis*, 52(10):4731 – 4744.

[Marrel *et al.*, 2009] Marrel, A., Iooss, B., Laurent, B. et Roustant, O. (2009). Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety*, 94(3):742 – 751.

[Marzat *et al.*, 2012] Marzat, J., Walter, E. et Piet-Lahanier, H. (2012). Worst-case global optimization of black-box functions through kriging and relaxation. *Journal of Global Optimization*, 55(4):707–727.

[Matheron, 1963] Matheron, G. (1963). *Traité de géostatistique appliquée : Le Krigeage*. Numéro vol. 2 de Mémoires du Bureau de Recherches Géologiques et Minières. Éditions Technip.

[Miller *et al.*, 2010] Miller, F., Vandome, A. et John, M. (2010). *Fides (Reliability)*. VDM Publishing.

[Močkus, 1975] Močkus, J. (1975). On bayesian methods for seeking the extremum. *In Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer Berlin Heidelberg.

[Mohammadi *et al.*, 2015] Mohammadi, H., Le Riche, R. et Touboul, E. (2015). Making EGO and CMA-ES Complementary for Global Optimization. *In Learning and Intelligent Optimization*, volume Volume 8994 de *9th International Conference, LION 9, Lille, France, January 12-15, 2015. Revised Selected Papers*, pages pp 287–292.

[Moré, 1978] Moré, J. (1978). The Levenberg-Marquardt algorithm : Implementation and theory. *In* Watson, G. A., éditeur : *Numerical Analysis*, volume 630 de *Lecture Notes in Mathematics*, chapitre 10, pages 105–116. Springer Berlin Heidelberg.

[Morio, 2011] Morio, J. (2011). Influence of input {PDF} parameters of a model on a failure probability estimation. *Simulation Modelling Practice and Theory*, 19(10):2244 – 2255.

[Morio, 2012] Morio, J. (2012). Extreme quantile estimation with nonparametric adaptive importance sampling. *Simulation Modelling Practice and Theory*, 27:76 – 89.

[Morio et Balesdent, 2015] Morio, J. et Balesdent, M. (2015). *Estimation of Rare Event Probabilities in Complex Aerospace and Other Systems : A Practical Approach*. Elsevier Science.

[Morris, 1991] Morris, M. D. (1991). Factorial sampling plans for preliminary computational experiments. *Technometrics*, 33(2):161–174.

[Morris, 2012] Morris, M. D. (2012). Gaussian surrogates for computer models with time-varying inputs and outputs. *Technometrics*, 54(1):42–50.

[Nelder et Mead, 1965] Nelder, J. A. et Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, 7:308–313.

[Nguyen et Widrow, 1990] Nguyen, D. et Widrow, B. (1990). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. *In*

Neural Networks, 1990., 1990 IJCNN International Joint Conference on, pages 21–26. IEEE.

[Ong *et al.*, 2006] Ong, Y.-S., Nair, P. B. et Lum, K. Y. (2006). Max-min surrogate-assisted evolutionary algorithm for robust design. *Trans. Evol. Comp*, 10(4):392–404.

[Oussar, 1998] Oussar, Y. (1998). *Réseaux d'ondelettes et réseaux de neurones pour la modélisation statique et dynamique de processus.* Thèse de doctorat, UNIVERSITÉ PARIS VI.

[Padulo et Guenov, 2011] Padulo, M. et Guenov, M. D. (2011). Worst-case robust design optimization under distributional assumptions. *International Journal for Numerical Methods in Engineering*, 88(8):797–816.

[Paredis, 1995] Paredis, J. (1995). Coevolutionary computation. *Artificial life*, 2(4):355–375.

[Park et Sandberg, 1991] Park, J. et Sandberg, I. W. (1991). Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257.

[Parr, 2013] Parr, J. (2013). *Improvement criteria for constraint handling and multiobjective optimization.* Thèse de doctorat, University of Southampton.

[Pastel *et al.*, 2014] Pastel, R., Morio, J. et Le Gland, F. (2014). Improvement of satellite conflict prediction reliability through use of the adaptive splitting technique. *Proceedings of the Institution of Mechanical Engineers, Part G : Journal of Aerospace Engineering*, 228(1):77–85.

[Pedregosa *et al.*, 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. et Duchesnay, E. (2011). Scikit-learn : Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Peng et Wu, 2014] Peng, C.-Y. et Wu, C. F. J. (2014). On the choice of nugget in kriging modeling for deterministic computer experiments. *Journal of Computational and Graphical Statistics*, 23(1):151–168.

[Picheny, 2015] Picheny, V. (2015). Multiobjective optimization using gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing*, 25(6):1265–1280.

[Powell, 1998] Powell, M. (1998). Direct search algorithms for optimization calculations. *Acta Numerica*, 7:287–336.

[Prokhorov, 1956] Prokhorov, Y. V. (1956). Convergence of random processes and limit theorems in probability theory. *Theory of Probability & Its Applications*, 1(2):157–214.

[Pronzato et Müller, 2012] Pronzato, L. et Müller, W. G. (2012). Design of computer experiments : space filling and beyond. *Statistics and Computing*, 22(3):681–701.

[Raghuwanshi et Kakde, 2004] Raghuwanshi, M. M. et Kakde, O. G. (2004). Survey on multiobjective evolutionary and real coded genetic algorithms. *In Proceedings of the 8th Asia Pacific symposium on intelligent and evolutionary systems*, pages 150–161.

[Rajabi *et al.*, 2015] Rajabi, M. M., Ataie-Ashtiani, B. et Simmons, C. T. (2015). Polynomial chaos expansions for uncertainty propagation and moment independent sensitivity analysis of seawater intrusion simulations. *Journal of Hydrology*, 520:101 – 122.

[Rasmussen et Williams, 2005] Rasmussen, C. E. et Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

[Regis, 2016] Regis, R. G. (2016). Trust regions in kriging-based optimization with expected improvement. *Engineering Optimization*, 48(6):1037–1059.

[Rehman *et al.*, 2014] Rehman, S., Langelaar, M. et van Keulen, F. (2014). Efficient kriging-based robust optimization of unconstrained problems. *Journal of Computational Science*, 5(6):872 – 881.

[Riche *et al.*, 2001] Riche, R. L., Gualandris, D., Thomas, J. et Hemez, F. (2001). Neural identification of non-linear dynamic structures. *Journal of Sound and Vibration*, 248(2):247 – 265.

[Rockafellar, 2007] Rockafellar, R. T. (2007). Coherent approaches to risk in optimization under uncertainty. *Tutorials in operations research*, 3:38–61.

[Rockafellar et Uryasev, 2000] Rockafellar, R. T. et Uryasev, S. (2000). Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–41.

[Rockafellar et Uryasev, 2002] Rockafellar, R. T. et Uryasev, S. (2002). Conditional value-at-risk for general loss distributions. *Journal of banking & finance*, 26(7):1443–1471.

[Rubinstein et Kroese, 2004] Rubinstein, R. Y. et Kroese, D. P. (2004). *The Cross Entropy Method : A Unified Approach To Combinatorial Optimization, Monte-carlo Simulation (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Rumelhart *et al.*, 1986] Rumelhart, D. E., Hinton, G. E. et Williams, R. J. (1986). Parallel distributed processing : Explorations in the microstructure of cognition, vol. 1. chapitre Learning Internal Representations by Error Propagation, pages 318–362. MIT Press, Cambridge, MA, USA.

[Salama et Abdelbar, 2015] Salama, K. M. et Abdelbar, A. M. (2015). Learning neural network structures with ant colony algorithms. *Swarm Intelligence*, 9(4):229–265.

[Saltelli, 2002] Saltelli, A. (2002). Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280–297.

[Santner *et al.*, 2003] Santner, T. J., B., W. et W., N. (2003). *The Design and Analysis of Computer Experiments*. Springer-Verlag.

[Saporta, 2006] Saporta, G. (2006). *Probabilités, analyse des données et statistique*. Editions Technip.

[Schuëller et Jensen, 2008] Schuëller, G. et Jensen, H. (2008). Computational methods in optimization considering uncertainties – an overview. *Computer Methods in Applied Mechanics and Engineering*, 198(1):2 – 13. Computational Methods in Optimization Considering Uncertainties.

[Scott, 1992] Scott, D. (1992). *Multivariate Density Estimation : Theory, Practice, and Visualization*. A Wiley-interscience publication. Wiley.

[Siarry, 2014] Siarry, P. (2014). *Métaheuristiques*. Algorithmes (Paris). Eyrolles.

[Silverman, 1986] Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman & Hall.

[Snoek *et al.*, 2012] Snoek, J., Larochelle, H. et Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *In Advances in neural information processing systems*, pages 2951–2959.

[Sobol et Kucherenko, 2009] Sobol, I. M. et Kucherenko, S. (2009). Derivative based global sensitivity measures and their link with global sensitivity indices. *Math. Comput. Simul.*, 79(10):3009–3017.

[Sobol′, 2001] Sobol′, I. (2001). Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and Computers in Simulation*, 55(1–3):271 – 280. The Second {IMACS} Seminar on Monte Carlo Methods.

[Sontag, 1997] Sontag, E. (1997). Recurrent neural networks : Some systems-theoretic aspects. *In Dealing with Complexity : a Neural Network Approach*, pages 1–12. Springer-Verlag.

[Spall, 2005] Spall, J. (2005). *Introduction to Stochastic Search and Optimization : Estimation, Simulation, and Control*. Wiley Series in Discrete Mathematics and Optimization. Wiley.

[Srinivas *et al.*, 2010] Srinivas, N., Krause, A., Seeger, M. et Kakade, S. M. (2010). Gaussian process optimization in the bandit setting : No regret and experimental design. *In* Fürnkranz, J. et Joachims, T., éditeurs : *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1015–1022. Omnipress.

[Storn et Price, 1997] Storn, R. et Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359.

[Sudret, 2008] Sudret, B. (2008). Global sensitivity analysis using polynomial chaos expansions. *Reliability Engineering & System Safety*, 93(7):964 – 979. Bayesian Networks in Dependability.

[Suryawanshi et Ghosh, 2016] Suryawanshi, A. et Ghosh, D. (2016). Reliability based optimization in aeroelastic stability problems using polynomial chaos based metamodels. *Structural and Multidisciplinary Optimization*, 53(5):1069–1080.

[Svenson, 2011] Svenson, J. D. (2011). *Computer experiments : Multiobjective optimization and sensitivity analysis*. Thèse de doctorat, The Ohio State University.

[Taguchi et Phadke, 1989] Taguchi, G. et Phadke, M. S. (1989). *Quality Control, Robust Design, and the Taguchi Method*, chapitre Quality Engineering through Design Optimization, pages 77–96. Springer US, Boston, MA.

[Tarantola, 2004] Tarantola, A. (2004). *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

[Tibshirani, 1994] Tibshirani, R. (1994). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.

[Tissot et Prieur, 2012] Tissot, J.-Y. et Prieur, C. (2012). Bias correction for the estimation of sensitivity indices based on random balance designs. *Reliability Engineering and System Safety*, 107:205–213. Special Issue : SAMO 2010.

[Trebatický et Pospíchal, 2008] Trebatický, P. et Pospíchal, J. (2008). *Artificial Neural Networks - ICANN 2008 : 18th International Conference, Prague, Czech Republic,*

September 3-6, 2008, Proceedings, Part II, chapitre Neural Network Training with Extended Kalman Filter Using Graphics Processing Unit, pages 198–207. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Turlach, 1993] Turlach, B. A. (1993). Bandwidth Selection in Kernel Density Estimation : A Review. *In CORE and Institut de Statistique*, pages 23–493.

[Tutschku, 1995] Tutschku, K. (1995). Recurrent multilayer perceptrons for identification and control : The road to applications.

[Ulaganathan *et al.*, 2015] Ulaganathan, S., Couckuyt, I., Dhaene, T., Degroote, J. et Laermans, E. (2015). Performance study of gradient-enhanced kriging. *Engineering with Computers*, 32(1):15–34.

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.

[Viana *et al.*, 2009] Viana, F. A. C., Venter, G. et Balabanov, V. (2009). An algorithm for fast optimal Latin hypercube design of experiments. *International Journal for Numerical Methods in Engineering*.

[Wagner *et al.*, 2010] Wagner, T., Emmerich, M., Deutz, A. et Ponweiser, W. (2010). *Parallel Problem Solving from Nature, PPSN XI : 11th International Conference, Kraków, Poland, September 11-15, 2010, Proceedings, Part I*, chapitre On Expected-Improvement Criteria for Model-based Multi-objective Optimization, pages 718–727. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Wang et Shan, 2007] Wang, G. G. et Shan, S. (2007). Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical design*, 129(4):370–380.

[Wang et Huang, 2011] Wang, X. et Huang, Y. (2011). Convergence study in extended kalman filter-based training of recurrent neural networks. *IEEE Transactions on Neural Networks*, 22(4):588–600.

[Watkins, 1992] Watkins, A. (1992). On models of spatial covariance. *Computational Statistics and Data Analysis*, 13(4):473 – 481.

[Werbos, 1990] Werbos, P. J. (1990). Backpropagation through time : what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

[Wild et Shoemaker, 2013] Wild, S. M. et Shoemaker, C. A. (2013). Global convergence of radial basis function trust-region algorithms for derivative-free optimization. *SIAM Review*, 55(2):349–371.

[Wu et Chen, 2009] Wu, J. et Chen, E. (2009). *Advances in Neural Networks – ISNN 2009 : 6th International Symposium on Neural Networks, ISNN 2009 Wuhan, China, May 26-29, 2009 Proceedings, Part III*, chapitre A Novel Nonparametric Regression Ensemble for Rainfall Forecasting Using Particle Swarm Optimization Technique Coupled with Artificial Neural Network, pages 49–58. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Zhang et Li, 2007] Zhang, Q. et Li, H. (2007). Moea/d : A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731.

[Zhang *et al.*, 2010] Zhang, Q., Liu, W., Tsang, E. P. K. et Virginas, B. (2010). Expensive multiobjective optimization by moea/d with gaussian process model. *IEEE Trans. Evolutionary Computation*, 14(3):456–474.

[Zhao et Zhang, 2014] Zhao, P. et Zhang, T. (2014). Stochastic optimization with importance sampling. *arXiv preprint arXiv :1401.2753*.

[Zitzler *et al.*, 2000] Zitzler, E., Deb, K. et Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms : Empirical results. *Evol. Comput.*, 8(2):173–195.

[Zitzler *et al.*, 2001] Zitzler, E., Laumanns, M. et Thiele, L. (2001). SPEA2 : Improving the strength pareto evolutionary algorithm for multiobjective optimization. *In* Giannakoglou, K. C., Tsahalis, D. T., Périaux, J., Papailiou, K. D. et Fogarty, T., éditeurs : *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100. International Center for Numerical Methods in Engineering.