Gaussian process regression on nested subspaces

Thierry Gonon

Numéro d'ordre NNT : 2022LYSEC009

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

École Doctorale 512
École Doctorale InfoMaths

Spécialité de doctorat : Mathématiques et applications
Discipline : Mathématiques

Soutenue publiquement le 03/03/2022 par

Thierry GONON

Gaussian process regression on nested subspaces

Devant le jury composé de :

Mme Clémentine Prieur	Professeur, Université Grenoble Alpes	Président
M. Bertrand Iooss	Chercheur senior, EDF R&D	Rapporteur
M. Luc Pronzato	Directeur de recherche, CNRS	Rapporteur
M. Olivier Roustant	Professeur, INSA Toulouse	Examinateur
M. Bruno Demory	Ingénieur simulation CFD, Valeo	Invité
Mme Christophette Blanchet-Scalliet	Maître de conférence, Ecole Centrale de Lyon	Directeur
Mme Céline Helbert	Maître de conférence, Ecole Centrale de Lyon	Directeur

Remerciements

Bon, et bien je crois que le moment est venu. C'est un événement sacrément spécial que la fin d'une thèse, de trois années de recherche et d'approfondissement d'un sujet. Il est temps pour moi de remercier les principaux acteurs de la mienne, qui ont fait en sorte qu'elle se passe le mieux possible. Préparez-vous à la lecture des remerciements les plus longs de l'histoire des remerciements.

Merci aux membres du jury d'avoir accepté votre mission : Bertrand Iooss, Clémentine Prieur, Luc Pronzato, et Olivier Roustant. En particulier merci à Luc Pronzato et Bertand Iooss pour avoir accepté de rapporter ma thèse.

Merci à Céline, Christophette, et Bruno, mes très chers encadrants. Merci Céline pour ton énergie permanente, pour tes idées à foison, pour m'avoir accordé ta confiance en m'accompagnant en projet, en stage, puis en thèse. Merci pour ton aide bibliographique, ta connaissance dans le domaine, et ton investissement dans la communauté française de la quantification d'incertitude. Merci ensuite Christophette, pour avoir accepté de me superviser avec Céline, pour ton engagement dans ma thèse, ton ardeur à réfléchir aux questions que je t'ai posées, tes remarques toujours pertinentes et tes éclairs de génie, et nos partages de calculs et de réflexions qui m'ont passionné et ont été une vraie source de motivation dans ma thèse. Merci à Bruno pour m'avoir suivi toutes les deux semaines avec assiduité, pour ton enthousiasme sans égal envers ton travail qui me fait vouer un respect sans borne en ta personne. Merci pour toujours avoir gardé de vue les enjeux industriels, une vision globale qui nous a été extrêmement utile pour nous recadrer dans un contexte, et parfois nous réaligner sur les bons enjeux, lorsque nous nous perdions un peu dans des développements mathématiques.

Merci au labo de maths dans son ensemble, pour la bonne ambiance qui y règne et pour l'intégration des jeunes. En particulier, merci Isabelle pour ta gentillesse, ton sens du service, pour m'avoir aidé à m'y retrouver dans les méandres administratifs. Merci Francis, pour avoir réglé tous les soucis informatiques que j'ai rencontrés, et pour m'avoir conseillé des outils très utiles dans ma vie de tous les jours. Merci Martin pour avoir consacré du temps à m'expliquer le fonctionnement du cluster de Centrale et m'avoir aidé à me servir de ton code 1D. Merci à Benoit Fabrèges et Roland Denis, responsables du cluster de l'ICJ, vous que j'ai sollicités à de nombreuses reprises (probablement de façon excessive) et qui avez pris le temps de répondre à mes questions relatives au cluster ou même pour des problèmes de code (bien que cela dépasse le cadre de votre fonction). Cela m'a débloqué plusieurs fois et je vous en suis très reconnaissant.

Merci à ceux qui ont collaboré de près ou de loin à la thèse. Merci à Manuel et Nicolas, vous qui avez oeuvré côté Valeo pour la naissance de ma thèse, qui avez gardé un oeil sur moi tout au long de celle-ci, et que je ne remercierai jamais assez. Merci à l'équipe de CFD des systèmes thermiques de Valeo, à laquelle j'ai parfois été rendre visite durant ces trois ans. Merci Xavier Bay pour nous avoir accueillis aux Mines de Saint-Etienne, pour avoir longuement discuté avec nous de ma thèse, et défriché très pédagogiquement les portions de votre travail que nous avons utilisées. Merci à Olivier Roustant et Jean-Yves Welschinger pour avoir accepté d'être membres de mon comité de suivi, pour avoir surveillé le déroulement de ma thèse avec un vrai soucis de me faire réussir, et en particulier Olivier d'un point de vue scientifique pour nous avoir apporté votre expertise dans le domaine et nous avoir proposé des pistes.

Un grand merci aux membres de la vénérable cathestr, JC, Paypouze, Jacky, et Hippo, pour m'avoir accueilli dans votre coloc les bras ouverts au début de ma thèse. En particulier, merci à toi Hippolyte, à qui j'ai coûtume de dire que je dois ma thèse. Merci d'avoir été me chercher pour faire un projet de recherche et m'avoir fait connaître Céline par la même occasion.

Un merci tout particulier à mes chers cobureaux, avec qui j'ai eu le plaisir de passer mes journées, et parfois aussi mes soirées ou week end. Merci à Mona, tout d'abord, pour avoir passé avec moi mes premières longues journées d'hiver au labo, pour ta gentillesse à toute épreuve, et pour ta formidable réactivité au dooble. Merci à Nicolas, ensuite, toi qui dès le premier jour m'a demandé : "Dis-moi Thierry, est-ce que tu sais jouer au tarot ?". Merci pour les footings dans les pentes de la Croix-Rousse que tu m'as emmené faire pour mon plus grand plaisir (même si je n'avais pas le niveau), pour la sortie ski à laquelle tu m'as convié et que je qualifierais de journée des descentes de l'extrême. Merci à Laura, ou Madame Tatouille, notre mère à tous bien sûr, une source d'inspiration pour beaucoup de choses, pour avoir contribué à l'ambiance du labo avec les regrettés vendredis pâtisserie, pour le sujet de stage que tu as fait avec Mathilde et que je vous ai honteusement piqué, pour toutes les activités que tu nous as organisées : balades, escape game, et j'en passe, pour mon article et ma présentation au séminaire des doctorants que tu as relus avec une minutie qui m'a impressionné, dont je n'aurais pas été capable pour d'autres. Merci à Angèle, toi qui est la cobureau avec laquelle j'ai passé le plus de temps pendant la thèse, merci pour les problèmes mathématiques que tu nous as exposés et auxquels nous avons réfléchis devant le tableau, ton éthique de travail et ta conscience professionnelle qui je pense m'ont tiré vers le haut, merci pour ton incroyable énergie, enrhumée ou non, pour avoir été la précurseure des aller-retour à vélo, et la plus assidue je le reconnais volontiers, merci pour les footings sur le temps du midi, pour tous les jeux que nous avons faits quand nous n'étions que deux ou avec d'autres (peut-être rattrapera-t-on un jour le score de Laura et Mathilde au hanabi), pour enfin nous avoir invités à ton magnifique mariage. Merci à Mathilde, informaticienne pour la thèse mais mathématicienne dans le coeur, merci pour ta bienveillance envers moi, j'ai vraiment apprécié ta compagnie, nos discussions en voiture après les quelques soirées que nous avons faites ensemble, ainsi que notre visionnage de Forest Gump en temps de grève qui restera inoubliable, je me réjouis grandement de ton heureux événement récent avec tout de même un léger pincement au coeur qu'il ne soit pas arrivé le 23 septembre, histoire que ce jour soit définitivement éligible à la fériérisation. Merci Mélina, mon illustre prédécesseure de thèse, pour avoir pris le temps de répondre à mes questions à certains moments de ma thèse quand j'en ai eu besoin, qui ont été charniers dans le déroulement de celle-ci et qui ont notamment conduit à des choix sur le format de mes codes R largement inspirés des tiens. Merci également pour les quelques fois où je t'ai rencontrée dans un cadre plus festif où j'ai pu découvrir ton sens de l'humour dévastateur qui fait mouche sur moi à tous les coups. Merci à Antoine, avec qui j'ai de commun nos années d'élèves ingénieurs, merci pour nous avoir fait découvrir ce jeu exceptionnel qu'est le hanabi, qui est devenu une véritable affaire d'état, vecteur de rapports passionnés sur le temps du midi, merci pour avoir assisté en ma compagnie et celle de Reda à la dernière grande victoire de Roger Federer (une demi-finale de Wimbledon d'anthologie contre Nadal), merci également pour m'avoir inclus au groupe des restos avec Lucas et Antonio. Merci à vous trois pour avoir traversé nos thèses côte à côte, pour nos soirées de débats mathématiques, politiques, sociétaux et sportifs enfiévrés. Merci Thibault pour avoir accepté les yeux fermés de faire le RAID avec moi, pendant lequel on s'est je pense bien marrés, nous les MItigés, les combattants, les talentueux, ou, pour reprendre tes mots, les chaudières. Merci à Hugues, toi qui as une place particulière en tant

que dernier rescapé des compagnons du flunch à être resté dans l'enceinte de l'école, merci pour les soirées de réflexion intense autour d'énigmes mathématiques, et pour les parties endiablées de race for the galaxy (moi l'élève a encore du travail pour rattraper le niveau du maître). Merci Reda, le célèbre auteur de la fonction de Reda, comme aime à l'appeler Noé, merci pour ta tranquillité en toute circonstance, pour tes compétences avancées de recherche bibliographique, pour les confs que j'ai passées à tes côtés, ce qui m'a permis de me sentir moins seul, enfin pour notre goût commun du foot et les parties que nous avons jouées ensemble (une technique de jeu et un contrôle de balle qui méritent tout mon respect). Merci enfin à ceux qui ont fait un bout de chemin avec moi en se rajoutant à la formidable bande : Benjamin, Tania, Nicolas (le post-doc), Matthieu, Josselin et bien sûr la prometteuse relève : Noé et Benoit.

Un grand merci à la compagnie élargie du flunch : Hugues en premier artilleur, Natan le plus sympathique des malchanceux, Thomas saboteur dans l'âme, Thibault digne héritier d'Audiard et de Coluche, Clément frère de maillot (allez Paris) et meilleur binôme en 1A, Antonio le plus français des espagnols et solide compère des TD de la mort en 3A. Notre esprit de groupe a perduré malgré la perte tragique de notre vénéré flunch, sa disparition nous a rendus plus forts et plus soudés et je vous remercie pour les nombreux skype que nous avons faits, pour les parties acharnées de pictionnary, et pour les parties de saboteur qui auront révélé une absence totale de mauvaise fois au sein du groupe, sans aucune réserve possible.

Un mot sur ma famille qui a joué un rôle important. Merci papy pour m'avoir transmis ton goût des maths, de leur histoire (que je ne connais que trop peu et j'essaie d'y remédier), les livres que tu m'as passés, et pour m'avoir fait découvrir Mickael Launay. Merci à toi et mamie pour nos appels parfois longs au téléphone les week end, qui me tiennent compagnie. Merci papa et maman pour m'avoir soutenu dans ma décision audacieuse de me lancer dans un doctorat. Merci à vous ainsi que Stéphane et Laura. Merci Stéphane pour les crises de fou rire que tu déclenches chez moi, pour nos nombreux délires qui désespèrent notre entourage, pour tes anecdotes croustillantes et hors du commun, pour ta vie sociale riche et saine qui m'inspire dans mes propres relations. Oui c'est le petit frère qui sert d'exemple au grand. Merci Laura, toi qui sera toujours tendrement "ma petite Laura" même si tu me répètes souvent que tu n'as plus 8 ans tout de même, experte reconnue en films de noël, que je vois grandir et s'épanouir à vue d'oeil. Si tu peux me rendre un service, ça serait de ne jamais t'arrêter de rire, car c'est ma plus grande source de joie. Même si à mon grand regret, je vous ai dégouté des maths, j'épie avec curiosité la direction que prendront vos études qui seront de toute façon nouvelles pour la famille et source d'ouverture vers d'autres domaines. Merci papa pour ton entrain, ta verve, tes anecdotes de pêche, toutes les vacances que tu nous organises et qui sont de véritables bulles d'air pour moi. Merci ma petite maman, pour ta tendresse sans faille, ton soucis de notre bien-être permanent et ton goût pour la lecture dont j'essaie de m'inspirer. Merci Michèle et Jojo pour votre compagnie toujours appréciée et pour avoir occupé mon confinement avec vos superbes cours de danse en ligne. Merci à toute ma famille dans son ensemble, et en particulier la family tribe, Hélène, Pascal, Benoît, Claire, Geoffroy, Lié, et surtout Philippe, toi qui a pris du temps pour relire mon article avec soin en faisant valoir ton expertise de la grammaire anglaise.

Je remercie pour finir trois professeurs qui m'ont profondément marqué et ont également grandement influencé ma scolarité : d'abord Monsieur Chériaux (avec un x parce qu'il y en a plusieurs), mon formidable professeur de CM1-CM2, un des hommes les plus cultivés et passionnants qu'il m'ait été donné de rencontrer, ensuite Monsieur Consigli qui a laissé une

trace indélébile sur mes années collège. Je dirais que si Monsieur Chériaux m'a fait aimé l'école, Monsieur Consigli m'a fait aimé les maths. Merci également à Monsieur Jaffrenoux, qui a révolutionné mon année de terminale, qui m'a absolument convaincu de faire une classe prépa, ce qui a indirectement conduit à ma thèse.

Je termine cette première et remuante expérience des remerciements avec une petite larme à l'oeil et souhaite à présent une bonne lecture de la suite aux plus courageux d'entre vous.

Résumé

Les métamodèles sont très largement utilisés dans l'industrie pour prédire la sortie des codes de calcul coûteux. Comme ces codes de calcul font intervenir une grande quantité de variables d'entrée, créer directement un grand métamodèle dépendant de l'ensemble des entrées apparait trop ambitieux. Les industriels choisissent par conséquent de procéder séquentiellement. Ils réalisent des études en plusieurs étapes avec des métamodèles se concentrant sur des ensembles de variables de plus en plus grands. Les variables non prises en compte sont fixées à une valeur nominale. La dimension de l'espace des entrées grandit à chaque étape. Cependant, l'information obtenue aux étapes précédentes est perdue car un nouveau plan d'expérience est généré pour construire le métamodèle. Dans cette thèse, une approche alternative est introduite, utilisant tous les plans d'expériences générés depuis le début plutôt que seulement celui de l'étape en cours. Ce métamodèle utilise la régression par processus Gaussiens et est appelé seqGRP (sequential Gaussian process regression). A chaque étape, la sortie est modélisée par la somme de deux processus : le processus qui modélisait la sortie à l'étape précédente et un processus correctif. Le premier est défini sur le sous-espace d'entrée de l'étape précédente tandis que le deuxième est défini sur le sous-espace de l'étape en cours. Le processus correctif représente l'information apportée par les variables libérées à l'étape concernée. Il a la particularité d'être nul sur le sous-espace de l'étape précédente pour assurer la cohérence de la modélisation entre les étapes. Premièrement, des candidats pour les processus correctifs sont proposés, qui ont la particularité d'être nuls sur un continuum de points. Ensuite, un algorithme d'EM (Expectation-Maximization) est implémenté pour estimer les paramètres des processus. Enfin, le métamodèle seqGPR est comparé à un métamodèle de krigeage classique qui modélise la sortie par un processus Gaussien stationnaire. La comparaison est faite sur deux exemples analytiques, un en deux étapes allant jusqu'à la dimension 4, un autre en trois étapes allant jusqu'à la dimension 15. La méthodologie introduite est également évaluée sur un exemple industriel allant de la dimension 11 à la dimension 15. Dans tous ces cas test, le métamodèle seqGPR a de meilleures performances, ou tout du moins est aussi bon que le krigeage. En parallèle, une méthodologie est proposée pour construire les échantillons d'entraînement du métamodèle. Enfin, deux problèmes complémentaires sont abordés : la présence de plusieurs plans d'expérience sur différents sous-espaces à chaque étape, et l'enrichissement des plans d'expérience.

Mots clés : Métamodèle, Krigeage, Régression par processus Gaussiens, Grande dimension, Conditionnement infini, Multifidélité, Espaces imbriqués, Problèmes d'espace d'entrées de taille variable.

Abstract

Metamodels are widely used in industry to predict the output of an expensive computer code. As industrial computer codes involve a large amount of input variables, creating directly one big metamodel depending on the whole set of inputs may be a very challenging problem. Industrialists choose instead to proceed sequentially. They build metamodels depending on nested sets of variables (the variables that are set aside are fixed to nominal values), i.e. the dimension of the input space is progressively increased. However, at each step, the previous piece of information is lost as a new Design of Experiment (DoE) is generated to learn the new metamodel. In this thesis, an alternative approach is introduced, based on all the DoEs rather than just the last one. This metamodel uses Gaussian process regression and is called seqGPR (sequential Gaussian process regression). At each step, the output is supposed to be the realization of the sum of two independent Gaussian processes. The first one models the output at the previous step. It is defined on the input space of the previous step which is a subspace of the one of the current step. The second Gaussian process is a correction term defined on the input space of the current step. It represents the additional information provided by the newly released variables. The correction term has the particularity of being null on the subspace of the previous step so that there is a coherence between the steps. Firstly, some candidate Gaussian processes for the correction terms are suggested, which have the property of being null on an infinite continuous set of points. Then, an EM (Expectation-Maximization) algorithm is implemented to estimate the parameters of the processes. Finally, the metamodel seqGPR is compared to a classic kriging metamodel where the output is assumed to be the realization of one second order stationary Gaussian process. The comparison is made on two analytic examples, a first one with two steps, up to dimension 4, and a second one with three steps, up to dimension 15. The introduced methodology is also tested on an industrial example which goes from dimension 11 to dimension 15. In all these test cases, seqGPR performs better than, or at least as well as kriging. A methodology is suggested to build the samples used for the seqGPR metamodel. Two complementary issues are tackled: the presence of multiple designs in different subspaces at each step, and the enrichment of the training samples.

keywords: Metamodel, Kriging, Gaussian process regression, High dimension, Infinite conditioning, Multifidelity, Nested spaces, Variable-size design space problems.

Contents

Chapter 1

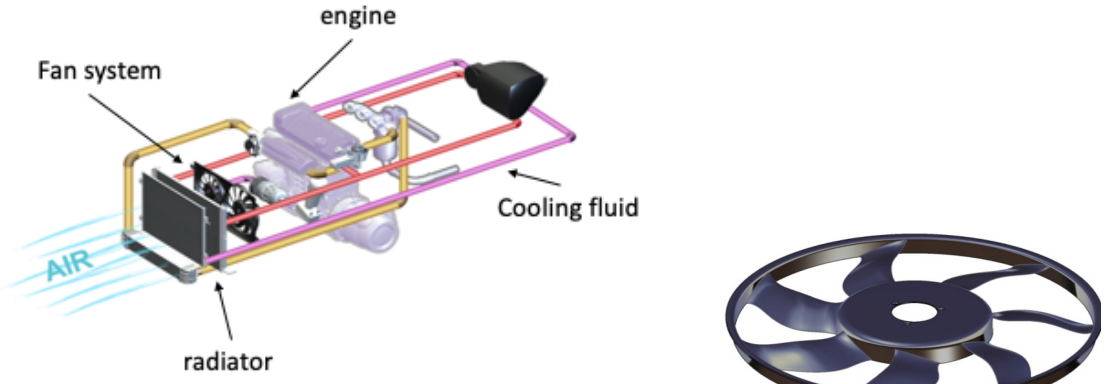
Introduction

This thesis is part of a collaboration between Institut Camille Jordan and Valeo. This collaboration was originally materialized by the ANR project PEPITO (Plans d'Experience Pour l'Industrie du Transport et l'Optimisation) which led to a first paper [Henner et al., 2019] (see appendix 9.5.1) about enrichment strategies for metamodels. A part of the thesis work has been the implementation of R routines using the package RcppArmadillo, and the submission of another paper [Gonon et al., 2021] (see appendix 9.5.2).
This chapter describes the industrial problem which motivates the work (see section 1.1), the choosen model to answer the problem (see section 1.2) and the issues that it creates (see section 1.3).

1.1 Industrial motivation

In Valeo, numerical codes are used to model physical phenomena involved in manufacturing. The industrial product which motivates this thesis work is the fan system. It is part of the car engine cooling system (see figure 1.1). This cooling system is composed of a cold fluid circulating in part in the car engine to regulate its temperature and in part in a radiator where it is itself cooled down. When the car moves, the wind generated by the car speed and reaching the radiator is sufficient to evacuate the heat from the fluid. When the car is motionless, the fan is activated to replace the wind.

Computer experiments make the search for performance easier and cheaper than physical experiments. However, computer codes are confronted to some limitations. As they model complex physical phenomena, they are often computationally expensive. One simulation of the computer code modelling the aerodynamical outputs of the fan system lasts 1h to 1h30 on 256 CPU. For the accoustical outputs, the simulation goes to 3 to 5 days on 1500 CPU. This problem of expensive time duration is solved by the use of metamodels [Forrester et al., 2008], [Sacks et al., 1989]. A metamodel is a simpler statistical model, like radial basis neural network [Cheng and Titterington, 1994], kriging model [Santner et al., 2003a], [Roustant et al., 2012], support vector regression [Clarke et al., 2005]. This simpler model is fit on a sample of well-chosen runs, also called design of experiments (DoE) [Pronzato and Müller, 2012], [Dupuy et al., 2015]. Metamodels provide fast approximations of the code outputs. Besides, they have a proactive role as they help to select relevant simulations to be run [Jones et al., 1998]. The computer code is then used more efficiently. Therefore, to study the fan system, Valeo experts use a mixed approach which combines the use of a computational fluid mecanics (CFD) computer code with designs of experiments and metamodels.



(a) Car engine cooling system

(b) Fan system

Figure 1.1: On left panel, diagram of the car engine cooling system. On right panel, diagram of fan system. [Valeo,]

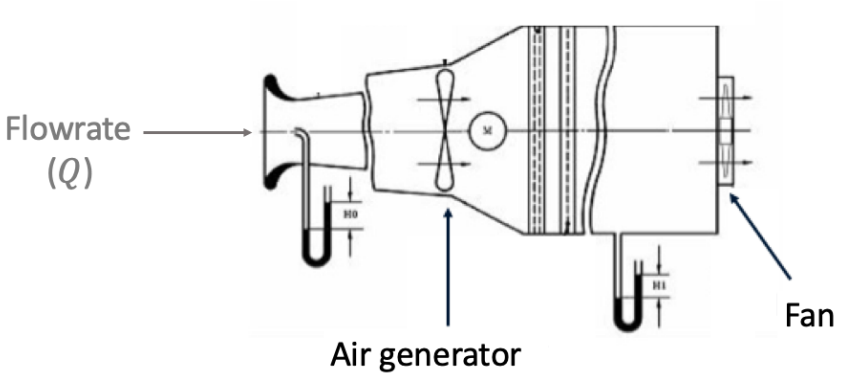
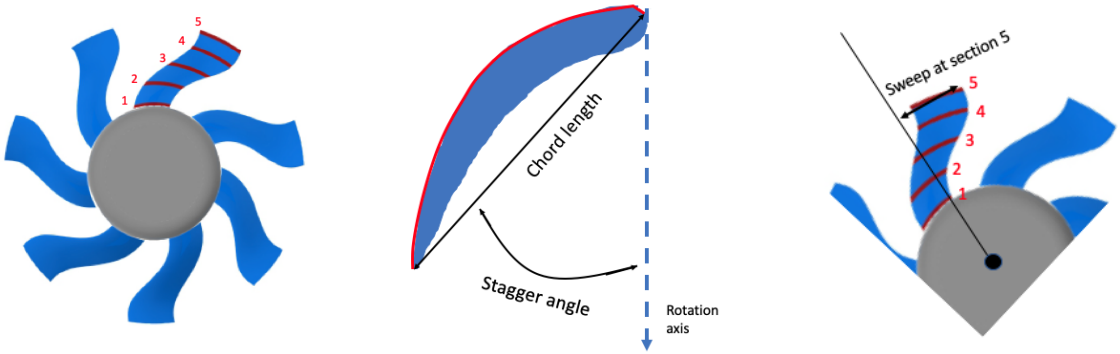


Figure 1.2: Visualization of the environment in which is modeled the fan system in the simulation code. The fan system is isolated from the engine cooling system. An air generator imposes a flowrate upstream from the fan. [Valeo,]

Building surrogate models can be difficult when numerical codes involve lots of variables. The output considered in this example is the Pressure difference (ΔP) between the upstream and the downstream of the air flow crossing the fan. The fan system is modeled independently from the cooling system context, as shown in figure 1.2. Among the variables studied on the fan system, one input is the flowrate (Q), which is, in this context, not delivered by the fan itself, but imposed upstream from it. The others are of geometric nature. The fan blade geometries are supposed identical to each other. The geometry is monitored at 5 different sections (see figure 1.3a). At each section, the stagger angle and the chord length are controlled (see figure 1.3b). The chord is the line formed by the two borders of the blade at the considered section. The stagger angle is the angle between the chord and the rotation axis. The sweep is manipulated at each section and shown in panel 1.3c. It is defined as the distance between the black line, formed by the rotation axis and the leading edge (right border on the figure) at the first section, and the leading edge at the considered section. At section 1, it is always equal to 0 and therefore not kept as an input. Other variables are involved like thickness, or cambers, more sections can be defined etc.



(a) Sections

(b) Chord length and stagger angle at a given section

(c) Definition of the sweep at section 5

Figure 1.3: On the left panel, visualization of the five sections of the blade. On the middle panel, definition of the stagger angle and the chord length at a given section. On the right panel, definition of the sweep at section 5.

Dimension reduction is a common approach to deal with a great number of variables. It consists in considering only the influential inputs. The influential inputs are often detected by sensitivity analysis methods ([Saltelli et al., 2008],[Sobol, 1993],[Iooss and Prieur, 2019]). However, this approach leads to a vicious circle. Indeed, on one side, the variables selected by sensitivity analysis are relevant only if the metamodel is sufficiently accurate, and on the other side, the dimension reduction is used to increase the metamodel accuracy.

Valeo experts chose an alternative method that consists in a study of several steps, which was defined in the PEPITO ANR project. The first step focuses on a small amount of variables deemed as important based on expert knowledge. The other variables are set to nominal values. A metamodel is created and used only for these variables until the understanding of their influence is sufficient. Then, to obtain better performances, new steps are proceeded with, in which new variables are added progressively. For example, in the case of the fan system, a first step of the study is done only on the flowrate, staggers and chords. Then, the second step also takes into account the sweeps. Further steps can be carried out with the thicknesses, cambers etc. and so on up to 30 variables.

The classical metamodeling approach for this kind of problem is to build a metamodel on a new independent space-filling design in the wider input space at each new step of the study. Yet, this approach may be criticized as the information provided by the previous DoE's and metamodels is disregarded. The question is then, what metamodel can be built to use optimally these ancient training samples. The idea is to use the information brought by the designs of all steps to provide a richer metamodel than the metamodel built on the design of the last step only.

1.2 Model

In the present work, an original metamodel is introduced, which takes into account the information from all the steps. The chosen approach consists in creating dependent metamodels from one step to another. The metamodels are based on Gaussian process regression [Santner et al., 2003a] [Williams and Rasmussen, 2006] (see section 2.1 in chapter 2).

One way of taking into account the different designs could be to use a multifidelity model for which the levels are not defined on the same input space. That is what is done in [Hebbal et al., 2021] with deep Gaussian processes (see last paragraph of subsection 2.1.3 in chapter 2). Nevertheless, this model is really expensive to learn. Furthermore, the multifidelity context is not suitable since the runs obtained at the previous steps have the same accuracy level. Another way could be to define a virtual categorical input, equal to the number of the step, that influences the choice of the input variables to consider. [Pelamatti et al., 2021] defines a Gaussian process whose input space varies dynamically with this virtual categorical input. This modelization does not take into account the fact that the input sets are entwined from one step to another. This particularity can lead to a simpler modelization. The approach in this work, called seqGPR (sequential Gaussian process regression), is based on a recursive statistical model inspired by the autoregressive multifidelity model [Kennedy and O'Hagan, 2000] (see subsection 2.1.3 in chapter 2) but with the same accuracy for all the runs and with an input space of increasing dimension.
The suggested metamodel is based on a probabilistic model of the function to approximate. It involves Gaussian processes built on nested subspaces and connected by additive relations. At a given step, the Gaussian process modeling the output on the current subspace is equal to the sum of the Gaussian process of the previous step, defined on a smaller subspace, and a correction process. As the two processes model the same function, they must coincide on the smaller subspace from the previous step. Therefore, the correction process must be null on the input subspace of the previous step, which is an infinite continuous set of points

1.3 Problematics and plan of the thesis

As regards the probabilistic model described in the previous section, one first difficulty is to build an appropriate correction term, null on a continuum of points with a numerically computable covariance kernel. One idea could be to enjoin the prediction to verify the nullity property. For example, relevant points from the concerned subspace can be added to the DoE. This technique is used in [Da Veiga and Marrel, 2012] in the context of monotonicity, boundedness, convexity constraints. One drawback is the greediness of that method. Besides, the nullity cannot be verified in the whole subspace. The correction process could be defined in a finite dimensional way, with adequate basis functions, as it is done in [Maatouk and Bay, 2017], [Lopera, 2019], [Bachoc et al., 2020], to ensure some properties of monotonicity, boundedness, convexity. Then the prediction could be computed by a Monte-Carlo scheme based on simulations using rejection sampling. Yet, this approach is greedy as the number of basis functions increases with the dimension. Instead of establishing the nullity in retrospect, it can be verified at first glance, as an intrinsic property of the correction term. In this thesis, the Gaussian process introduced in [Gauthier and Bay, 2012b], which uses an extension of the conditional expectation to an infinite continuous set of points, is retained. A tractable kernel is sought for this process which is then compared to other candidates verifying the nullity condition.

Once a tractable candidate for the correction term is proposed, another difficulty lies in the estimation of the hyperparameters. One way is to optimize the likelihood. This task can be numerically difficult, since the likelihood involves several sets of parameters for the different processes. One way to reduce the dimension of the optimization space is to propose nested designs as it is done in [Le Gratiet and Garnier, 2014]. In this thesis an alternative based on the expectation maximization algorithm [Friedman et al., 2001] is introduced. It allows the reduction of the dimension with limited constraints on the designs.

Special designs must be generated to be used by the seqGPR metamodel. At a given step, the design must be space-filling in the current subspace and be distant from the previous subspace, in order for the covariance matrices at stake to be invertible. A methodology must be defined to built such DoE's, inspired from the simulated annealing algorithm optimizing an LHS design for a given space-filling criterion (as in [Morris and Mitchell, 1995]), and taking into account the constraints proper to the metamodel seqGPR.

Chapter 2 defines the literature tools on which is based the method. The definition of the probabilistic model and the issue of building the correction processes null on infinite continuous sets of points are tackled in chapter 3. The estimation of the model parameters and the evaluation of the seqGPR method on analytical and industrial test cases are detailed in chapter 4. Adequate sampling strategies are developed in chapter 5. Chapter 6 describes some enhancements tried on the method.

Chapter 2

State of the art

This chapter details the different tools used in this thesis. Section 2.1 presents the kriging metamodel (or Gaussian process regression metamodel), from its definition to the associated results and methodologies. Section 2.2 lists the different answers existing in literature to impose some constraints on the kriging metamodel.

2.1 Generalities on the kriging metamodel

The present section defines the kriging metamodel as it is defined in [Williams and Rasmussen, 2006] and [Santner et al., 2003a]. First, some results about Gaussian vectors and Gaussian processes are recalled, that led to the definition of the kriging metamodel. Then, the parameter estimation by maximum likelihood is explained, and the EM algorithm (Expectation-Maximization) is detailed. The Multifidelity model is presented. The Sobol index used in sensibility analysis is defined. Finally, the usual sampling strategies associated with a metamodel are developped.

2.1.1 From Gaussian vectors to Gaussian process regression

Here are first described some theoretical results about Gaussian Vectors that are used in a kriging context.

Definition 1 (Equality in $L^2(\Omega)$) *Let $(\Omega, \mathcal{F}, \mathbb{P})$ denote a probability space. $L^2(\Omega)$ is the set of all random variables X defined on Ω such that $\mathbb{E}[X^2] < +\infty$. Two variables $X, Y \in L^2(\Omega)$ are said equal in $L^2(\Omega)$ ($X \overset{L^2(\Omega)}{=} Y$) if $\mathbb{E}[(X-Y)^2] = 0$.*

In the following, every variable is defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and all equalities between variables are equalities in L^2. The notation $X \overset{L^2(\Omega)}{=} Y$ is simplified in $X = Y$.

Definition 2 (Gaussian Vector) *A random vector $\boldsymbol{Y} = \begin{pmatrix} Y_1 & \cdots & Y_n \end{pmatrix}'$ is a Gaussian vector if every linear combination of its components is Gaussian, i.e. every $\sum_{i=1}^{n} \lambda_i Y_i$ (with $\lambda_i \in \mathbb{R}$) is a Gaussian Variable.*

Definition 3 (Conditional expectation) *Let X (resp. $\boldsymbol{Y}$) be a Gaussian variable (resp. Gaussian vector). The expectation of X conditioned by $\boldsymbol{Y}$, denoted by $\mathbb{E}[X \mid \boldsymbol{Y}]$ is the random variable U solution of*

$$\min_{U \in span(1, \boldsymbol{Y}')} \mathbb{E}\left[(X-U)^2\right],$$

where span$(1, \boldsymbol{Y}')$ denotes the set of all linear combinations of $(1, \boldsymbol{Y}')$. In particular, if X and $\boldsymbol{Y}$ are centered, the conditional expectation is the orthogonal projection of X in span$(\boldsymbol{Y})$. The definition of the conditional expectation can be extended if X is a vector by applying it component by component.

Proposition 1 (Explicit formula of the conditional expectation) *Let $\boldsymbol{V}$ be a Gaussian vector split in two subvectors $\boldsymbol{V} = \begin{pmatrix} \boldsymbol{V_1} & \boldsymbol{V_2} \end{pmatrix}$. The mean vector and covariance matrix of $\boldsymbol{V}$ can be defined by blocks:*

$$\begin{cases} \mathbb{E}[\boldsymbol{V}] & = & (\boldsymbol{m_1}, \boldsymbol{m_2})' \\ Cov(\boldsymbol{V}, \boldsymbol{V}) & = & \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \end{cases}$$

with $\boldsymbol{m_1} = \mathbb{E}[\boldsymbol{V_1}]$, $\boldsymbol{m_2} = \mathbb{E}[\boldsymbol{V_2}]$, $\Sigma_{11} = cov(\boldsymbol{V_1}, \boldsymbol{V_1})$, $\Sigma_{22} = cov(\boldsymbol{V_2}, \boldsymbol{V_2})$, $\Sigma_{12} = cov(\boldsymbol{V_1}, \boldsymbol{V_2})$, $\Sigma_{21} = cov(\boldsymbol{V_2}, \boldsymbol{V_1})$. The expectation of $\boldsymbol{V_1}$ conditioned by $\boldsymbol{V_2}$ is equal to

$$\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}] = \boldsymbol{m_1} + \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{V_2} - \boldsymbol{m_2}),$$

with Σ_{22}^{+} the Moore-Penrose generalized inverse of Σ_{22}. In particular, if Σ_{22} is positive definite, $\Sigma_{22}^{+} = \Sigma_{22}^{-1}$. It is a Gaussian vector of mean and covariance matrix:

$$\begin{cases} \mathbb{E}[\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]] & = & \boldsymbol{m_1}, \\ Cov(\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}], \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]) & = & \Sigma_{12}\Sigma_{22}^{+}\Sigma_{21}. \end{cases}$$

Proof *A proof of this proposition, entirely rewritten by hand, is suggested in appendix 9.2.1, as it was not encountered in the literature for the general case (this proposition is usually proven for the case Σ_{22} positive-definite).* ∎

Proposition 2 (Conditional vector) *Following the notations of proposition 1, the vector $[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}]$ is equal to*

$$[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}] = \boldsymbol{m_1} + \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{v} - \boldsymbol{m_2}) + V_1 - \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}].$$

It is a Gaussian vector of mean and covariance matrix given by

$$\begin{cases} \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}] & = & \boldsymbol{m_1} + \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{v} - \boldsymbol{m_2}) \\ cov([\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}], [\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}]) & = & \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{+}\Sigma_{21} \end{cases}$$

Proof *The proof of this proposition was not found in the literature and one written by hand is suggested in appendix 9.2.2.* ∎

Proposition 3 (Multiple conditioning) *Let $(U, \boldsymbol{V}, \boldsymbol{W})$ be a Gaussian vector with U (resp. $\boldsymbol{V}$ and $\boldsymbol{W}$) a Gaussian variable (resp. Gaussian vectors). Let denote by $X = [U \mid \boldsymbol{V} = \boldsymbol{v}]$ (resp. $\boldsymbol{Y} = [\boldsymbol{W} \mid \boldsymbol{V} = \boldsymbol{v}]$) the variable U (resp. $\boldsymbol{W}$) conditioned by $\boldsymbol{V} = \boldsymbol{v}$, then:*

$$[X \mid \boldsymbol{Y} = \boldsymbol{w}] = [U \mid \boldsymbol{V} = \boldsymbol{v}, \boldsymbol{W} = \boldsymbol{w}]$$

Proof *The proof of this proposition was not found in the literature and one written by hand is suggested in appendix 9.2.3* ∎

This last result on Gaussian vectors means that the conditioning can be carried out in any order, in one go or step by step. The next part is dedicated to Gaussian processes.

Definition 4 (Gaussian process) *A Gaussian Process $Y : \Omega \times \mathcal{X} \to \mathbb{R}$ verifies that for all $n \geqslant 1$, for all $x_1, \cdots, x_n$ in $\mathcal{X}$, $\begin{pmatrix} Y(x_1) & \cdots & Y(x_n) \end{pmatrix}$ is a Gaussian vector.*

Caracterization Y is entirely determined by its mean function $\mathbb{E}[Y(x)] = m(x)$ and its covariance kernel $cov(Y(x), Y(t)) = k(x,t)$.

Usual covariance kernels The following stationary tensor product kernels are often used in practice in the kriging context:

- Exponential kernel:
$$k(x,t) = \sigma^2 \prod_i \exp\left(-\frac{|x_i - t_i|}{\theta_i}\right)$$
- Matern $\frac{5}{2}$ kernel:
$$k(x,t) = \sigma^2 \prod_i \left(1 + \frac{\sqrt{5}\,|x_i - t_i|}{\theta_i} + \frac{5(x_i - t_i)^2}{\theta_i^2}\right) \exp\left(-\frac{\sqrt{5}\,|x_i - t_i|}{\theta_i}\right)$$
- Gaussian kernel:
$$k(x,t) = \sigma^2 \prod_i \exp\left(-\frac{(x_i - t_i)^2}{\theta_i^2}\right)$$

Definition 5 (Gaussian space) *Let $(Y(x))_{x\in\mathcal{X}}$ a Gaussian process indexed by $\mathcal{X}$. The Gaussian space engendered by Y is the closure of all linear combinations of the $Y(x)$ ($x \in \mathcal{X}$).*

Let $f : \mathcal{X} \to \mathbb{R}$ denote the output of an expensive computer code. A metamodel is used in order to approximate f. It is a simpler model quicker to evaluate. The kriging metamodel, or Gaussian process regression metamodel, is based on the assumption that the output f is the realization (a path) of a Gaussian Process Y. The Gaussian process is usually chosen second-order stationary: with constant mean m and stationary covariance kernel k (such that $k(x+h, t+h) = k(x,t)$). Then, given a sample of points $\mathbb{X} = \begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(n)} \end{pmatrix} \subset \mathcal{X}$ for which we know the output value ($f(\mathbb{X}) = \boldsymbol{y}$), the prediction mean and variance of the kriging metamodel are given by:

$$\left\{ \begin{array}{rcl} \hat{y}(x) & = & \mathbb{E}\left[Y(x) \mid Y(\mathbb{X}) = \boldsymbol{y}\right] \\ \hat{v}(x) & = & Var\left(Y(x) \mid Y(\mathbb{X}) = \boldsymbol{y}\right) \end{array} \right. \begin{array}{rcl} & = & m + k(x,\mathbb{X})k(\mathbb{X},\mathbb{X})^{-1}(\boldsymbol{y} - m\mathbf{1}_{\mathbb{X}}), \\ & = & k(x,x) - k(x,\mathbb{X})k(\mathbb{X},\mathbb{X})^{-1}k(\mathbb{X},x), \end{array} \tag{2.1}$$

where $\mathbf{1}_{\mathbb{X}}$ is the column vector full of ones of size equal to the size of $\mathbb{X}$. These formulae are a direct application of the formulae seen in proposition 1, considering the Gaussian vector $\begin{pmatrix} Y(x) & Y(\mathbb{X}) \end{pmatrix}'$.

2.1.2 Maximum Likelihood estimation and EM (Expectation - Maximization) algorithm

Knowing the distribution parameters of Y and some training data, the mean and variance of prediction of the corresponding kriging metamodel can be computed. But the parameters of Y are unknown a priori, so they first need to be infered from the training data. The parameter estimation can be done by maximum likelihood or cross-validation. In this thesis, the retained estimation method is the maximum likelihood. The method is described in this subsection, along with the EM algorithm which helps optimizing it in particular contexts.

Maximum Likelihood estimation The parameters of Y need to be estimated from the training data $Y(\mathbb{X}) = \boldsymbol{y}$. To do that, the likelihood, which is a function of the parameters η, is maximized. In rough terms, maximizing the likelihood can be interpreted as finding the set of parameters that maximizes the probability that the training data occurs. The likelihood taken at the set of parameters η (noted $\mathcal{L}(\eta; \boldsymbol{y})$) is equal to the density of the vector $Y(\mathbb{X})$ (assuming Y has the parameters η), applied at the observed values $\boldsymbol{y}$.

$$\mathcal{L}(\eta; \boldsymbol{y}) = h_{Y(\mathbb{X});\eta}(\boldsymbol{y}).$$

In the usual case, the output f is assumed to be a simulation of the Gaussian process $Y \sim \mathcal{GP}(m^*, (\sigma^*)^2 r_{\theta^*}(x, x'))$ of parameters $\eta^* = (m^*, (\sigma^*)^2, \theta^*)$. Denoting by $n_{\mathbb{X}}$ the number of points (rows) of $\mathbb{X}$, and by $\mathbf{1}_{\mathbb{X}}$ the unit column vector of size $n_{\mathbb{X}}$, the likelihood is equal to:

$$\mathcal{L}(\eta; \boldsymbol{y}) = \frac{1}{(2\pi\sigma^2)^{\frac{n_{\mathbb{X}}}{2}} \sqrt{|r_\theta(\mathbb{X},\mathbb{X})|}} \exp\left(-\frac{1}{2} \frac{(\boldsymbol{y} - m\mathbf{1}_{\mathbb{X}})' \, r_\theta(\mathbb{X},\mathbb{X})^{-1} (\boldsymbol{y} - m\mathbf{1}_{\mathbb{X}})}{\sigma^2}\right),$$

where $|r_\theta(\mathbb{X}, \mathbb{X})|$ denotes the determinant of the matrix $r_\theta(\mathbb{X}, \mathbb{X})$. Taking $-2\log(\mathcal{L})$ and keeping only the terms that depend on η, the quantity that needs to be minimized is:

$$l(\eta; \boldsymbol{y}) = n_{\mathbb{X}} \log(\sigma^2) + \log |r_\theta(\mathbb{X}, \mathbb{X})| + \frac{(\boldsymbol{y} - m\mathbf{1}_{\mathbb{X}})' \, r_\theta(\mathbb{X}, \mathbb{X})^{-1} (\boldsymbol{y} - m\mathbf{1}_{\mathbb{X}})}{\sigma^2}.$$

The optimal values of m and σ^2 are:

$$\begin{cases} m(\theta) = \dfrac{\mathbf{1}_{\mathbb{X}}' \, r_\theta(\mathbb{X}, \mathbb{X})^{-1} \, \boldsymbol{y}}{\mathbf{1}_{\mathbb{X}}' \, r_\theta(\mathbb{X}, \mathbb{X})^{-1} \, \mathbf{1}_{\mathbb{X}}} \\ \sigma^2(\theta) = \dfrac{(\boldsymbol{y} - m(\theta)\mathbf{1}_{\mathbb{X}})' \, r_\theta(\mathbb{X}, \mathbb{X})^{-1} (\boldsymbol{y} - m(\theta)\mathbf{1}_{\mathbb{X}})}{n_{\mathbb{X}}}. \end{cases}$$

Finally, finding (m, σ^2, θ) maximizing the likelihood is equivalent to finding θ minimizing:

$$l(\theta; \boldsymbol{y}) = n_{\mathbb{X}} \log\left(\sigma^2(\theta)\right) + \log |r_\theta(\mathbb{X}, \mathbb{X})| \,.$$

This optimization is done numerically.

EM algorithm The likelihood is often maximized numerically. This paragraph describes the EM algorithm as presented in [Friedman et al., 2001], which is an algorithm used to maximize the likelihood in the case where some observations are missing (which is of interest in the thesis).
The following notations are used:

- Z a random variable corresponding to the observed data and z, a scalar corresponding to its observation (which is known). The distribution of Z is parameterized by η^*.
- Z^m the missing data and z^m the corresponding observation (which is unkown).
- $T = (Z, Z^m)$ the complete data and $t = (z, z^m)$ the corresponding observation.

It is assumed that there exists a σ finite measure μ such that all T laws admit a density $f_T(t)$ with respect to μ. So Z admits a density $f_Z(z)$ with respect to μ. The loglikelihood of the observed data at η is given by:

$$l(\eta; z) = \log(h_{Z;\eta}(z)).$$

This loglikelihood can be difficult to optimize. The loglikelihood of the complete data is rather used, it is given by:

$$l_0(\eta; t) = \log(h_{T;\eta}(t)).$$

As t is not known entirely, the random variable $l_0(\eta; T) = \log(h_{T;\eta}(T))$ is used instead of $l_0(\eta; t)$. The expectation part of the EM algorithm consists in defining the expectation of this variable conditionally to $Z = z$ with the assumption that the Z distribution is parameterized by $\hat{\eta}$:

$$\mathcal{Q}(\eta, \hat{\eta}) = \mathbb{E}_{\hat{\eta}} \left[l_0(\eta, T) \mid Z = z \right]$$

The Maximization part of the EM algorithm consists in maximizing this quantity with respect to η.

- **Iteration 0**: Generate initial guess $\hat{\eta}^{(0)}$ for η.
- **Iteration** $k+1$: Compute $\hat{\eta}^{(k+1)}$ as solution of

$$\max_{\eta} \mathcal{Q}(\eta, \hat{\eta}^{(k)})$$

Proposition 4 *The likelihood increases at each iteration of the EM algorithm*

Proof *The proof, which is let in exercise in [Friedman et al., 2001], is detailed in appendix 9.2.4.* ∎

2.1.3 Multifidelity

This subsection describes the multifidelity metamodel in the context of Gaussian process regression (see [Kennedy and O'Hagan, 2000]). The multifidelity metamodel takes into account different levels of accuracy of the computer code. The low levels (resp. high levels) of accuracy are called low fidelity levels (resp. high fidelity levels), they are generally cheaper to evaluate (resp. more expensive to evaluate).

Metamodel For sake of simplicity, it is assumed that two levels of fidelity of the computer code are available. The low level is denoted by $f_1 : \mathcal{X} \to \mathbb{R}$, and the high level by $f_2 : \mathcal{X} \to \mathbb{R}$. The level of fidelity can result from the degree of refinement of the mesh in a finite element code, or the number of iterations in a Monte-Carlo based simulation.

f_1 (resp. f_2) is assumed to be the realization of a Gaussian process $Y_1 : \Omega \times \mathcal{X} \to \mathbb{R}$ (resp. $Y_2 : \Omega \times \mathcal{X} \to \mathbb{R}$). Y_2 is linked to Y_1 by the following formula:

$$Y_2(x) = g(x) Y_1(x) + Y_c(x), \ \forall x \in \mathcal{X}.$$

$g : \mathcal{X} \to \mathbb{R}$ is a known deterministic function and $Y_c : \Omega \times \mathcal{X} \to \mathbb{R}$ is a Gaussian process, independent of Y_1, and acting as a correcting term. The model is also valid if $g = \sum_i \beta_i f_i$ with $(\beta_i)_i$ parameters to estimate and $(f_i)_i$ deterministic functions.

The training data for the kriging metamodel is composed of one sample $\mathbb{X}_1 \subset \mathcal{X}$ on which the low fidelity level is observed $f_1(\mathbb{X}_1) = \boldsymbol{y_1}$, and another $\mathbb{X}_2 \subset \mathcal{X}$ on which the high fidelity level is observed $f_2(\mathbb{X}_2) = \boldsymbol{y_2}$. The mean and variance of prediction at $x \in \mathcal{X}$ of the multifidelity metamodel are the moments of the variable $[Y_2(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, Y_2(\mathbb{X}_2) = \boldsymbol{y_2}]$:

$$\begin{cases} \hat{y}(x) &= \mathbb{E}\left[Y_2(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, Y_2(\mathbb{X}_2) = \boldsymbol{y_2} \right] \\ \hat{v}(x) &= Var\left(Y_2(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, Y_2(\mathbb{X}_2) = \boldsymbol{y_2} \right) \end{cases}$$

Maximum likelihood estimation of the parameters The following notations are used:

- $\eta_1 = (m_1, \theta_1)$ denotes the set of parameters of $Y_1 \sim \mathcal{GP}\left(m_1, k_{\theta_1}(x, x')\right)$. $\eta_c = (m_c, \theta_c)$ denotes the set of parameters of $Y_c \sim \mathcal{GP}\left(m_c, k_{\theta_c}(x, x')\right)$. $\eta = (\eta_1, \eta_c)$ denotes the total set of parameters.
- n_1 and n_2 are respectively the sizes of $\mathbb{X}_1$ and $\mathbb{X}_2$.
- Σ_{θ_1} is the covariance matrix of Y_1 on $\mathbb{X}_1 \cup \mathbb{X}_2$. Σ_{θ_c} is the covariance matrix of Y_c on $\mathbb{X}_2$:

$$\left\{ \begin{array}{rcl} \Sigma_{\theta_1} & = & \begin{pmatrix} k_{\theta_1}(\mathbb{X}_1, \mathbb{X}_1) & k_{\theta_1}(\mathbb{X}_1, \mathbb{X}_2) \\ k_{\theta_1}(\mathbb{X}_2, \mathbb{X}_1) & k_{\theta_1}(\mathbb{X}_2, \mathbb{X}_2) \end{pmatrix} \\ \Sigma_{\theta_c} & = & k_{\theta_c}(\mathbb{X}_2, \mathbb{X}_2) \end{array} \right.$$

- $\mathbf{1}_n$ denotes the unit column vector of size n. $diag(u)$ denotes the diagonal matrix whose diagonal components are the components of the vector u. 0_{nm} is the null matrix with n rows and m columns.

If the designs are nested ($\mathbb{X}_2 \subset \mathbb{X}_1$), then $g(\mathbb{X}_2) \circ Y_1(\mathbb{X}_2) = \boldsymbol{u_1}$ is observed. The likelihood becomes

$$\begin{array}{rcll} \mathcal{L}(\eta; \boldsymbol{y_1}, \boldsymbol{y_2}) & = & h_{Y_1(\mathbb{X}_1), g(\mathbb{X}_2) \circ Y_1(\mathbb{X}_2) + Y_c(\mathbb{X}_2)}(\boldsymbol{y_1}, \boldsymbol{y_2}) & \\ & = & h_{Y_1(\mathbb{X}_1), \boldsymbol{u_1} + Y_c(\mathbb{X}_2)}(\boldsymbol{y_1}, \boldsymbol{y_2}) & \\ & = & h_{Y_1(\mathbb{X}_1), Y_c(\mathbb{X}_2)}(\boldsymbol{y_1}, \boldsymbol{y_2} - \boldsymbol{u_1}) & \\ & = & \underbrace{h_{Y_1(\mathbb{X}_1)}(\boldsymbol{y_1})}_{=\mathcal{L}_1(\eta_1; \boldsymbol{y_1})} \underbrace{h_{Y_c(\mathbb{X}_2)}(\boldsymbol{y_2} - \boldsymbol{u_1})}_{=\mathcal{L}_c(\eta_c; \boldsymbol{y_2} - \boldsymbol{u_1})} & \text{as } Y_1 \perp Y_c \end{array}$$

The likelihood is decoupled. η_1 and η_c can be estimated seperately by maximizing respectively $\mathcal{L}_1(\eta_1; \boldsymbol{y_1})$ and $\mathcal{L}_c(\eta_c; \boldsymbol{y_2} - \boldsymbol{u_1})$. Similarly, the prediction is also decoupled:

$$\left\{ \begin{array}{rcl} \hat{y}(x) & = & \mathbb{E}\left[Y_1(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}\right] + \mathbb{E}\left[Y_c(x) \mid Y_c(\mathbb{X}_2) = \boldsymbol{y_2} - \boldsymbol{u_1}\right] \\ \hat{v}(x) & = & Var\left(Y_1(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}\right) + Var\left(Y_c(x) \mid Y_c(\mathbb{X}_2) = \boldsymbol{y_2} - \boldsymbol{u_1}\right) \end{array} \right.$$

If $\mathbb{X}_1 \cap \mathbb{X}_2 = \emptyset$, neither $Y_1(\mathbb{X}_2)$ nor $Y_c(\mathbb{X}_2)$ is observed. This time, the prediction and the likelihood are not decoupled. The joint likelihood must be maximized:

$$\mathcal{L}(\eta; \boldsymbol{y_1}, \boldsymbol{y_2}) = h_{Y_1(\mathbb{X}_1), g(\mathbb{X}_2) \circ Y_1(\mathbb{X}_2) + Y_c(\mathbb{X}_2)}(\boldsymbol{y_1}, \boldsymbol{y_2})$$

[Zertuche, 2015] suggests to use an EM algorithm (see subsection 2.1.2) in this case.

Proposition 5 *In the case of multifidelity with not nested designs, the EM algorithm is the following*

- ***Iteration 0**: Generate initial guess $\hat{\eta}^{(0)} = (\hat{\eta}_1^{(0)}, \hat{\eta}_c^{(0)})$.*
- ***Iteration** $k+1$:*
  - *Compute $\hat{\eta}_1^{(k+1)}$ as solution of*

$$\max_{\eta_1} \mathcal{Q}_1(\eta_1, \hat{\eta}^{(k)})$$

  - *Compute $\hat{\eta}_c^{(k+1)}$ as solution of*

$$\max_{\eta_c} \mathcal{Q}_c(\eta_c, \hat{\eta}^{(k)})$$

with

$$\left\{\begin{array}{rcl}
\mathcal{Q}_1(\eta_1,\eta') &=& \frac{n_1+n_2}{2}\log(2\pi)-\frac{1}{2}\log(|\Sigma_{\theta_1}|)\\
&& -\frac{1}{2}tr\left(\left(k_{\theta_1}(\mathbb{X}_2,\mathbb{X}_2)-k_{\theta_1}(\mathbb{X}_2,\mathbb{X}_1)k_{\theta_1}(\mathbb{X}_1,\mathbb{X}_1)^{-1}k_{\theta_1}(\mathbb{X}_1,\mathbb{X}_2)\right)^{-1}\widetilde{\Sigma}_{\eta'}\right)\\
&& -\frac{1}{2}\left((\boldsymbol{y_1}-m_1\mathbf{1}_{n_1})' \quad \left(\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta'}}-m_1\mathbf{1}_{n_2}\right)'\right)\Sigma_{\theta_1}^{-1}\begin{pmatrix}\boldsymbol{y_1}-m_1\mathbf{1}_{n_1}\\ \widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta'}}-m_1\mathbf{1}_{n_2}\end{pmatrix}\\
\mathcal{Q}_c(\eta_c,\eta') &=& -\frac{n_2}{2}\log(2\pi)-\frac{1}{2}\log(|\Sigma_{\theta_c}|)\\
&& -\frac{1}{2}tr\left(\Sigma_{\theta_c}^{-1}diag(g(\mathbb{X}_2))\widetilde{\Sigma}_{\eta'}diag(g(\mathbb{X}_2))\right)\\
&& -\frac{1}{2}\left(\mathbb{Y}_2-g(\mathbb{X}_2)\circ\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta'}}-m_c\mathbf{1}_{n_2}\right)'\Sigma_{\theta_c}^{-1}\left(\mathbb{Y}_2-g(\mathbb{X}_2)\circ\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta'}}-m_c\mathbf{1}_{n_2}\right)
\end{array}\right.$$

and

$$\left\{\begin{array}{rcl}
\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta'}} &=& m_1'\mathbf{1}_{n_2}\\
&& +\left[\begin{array}{l}\left(k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_1)\quad k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_2)diag(g(\mathbb{X}_2))\right)\\ \times\begin{pmatrix}k_{\theta_1'}(\mathbb{X}_1,\mathbb{X}_1) & k_{\theta_1'}(\mathbb{X}_1,\mathbb{X}_2)diag(g(\mathbb{X}_2))\\ diag(g(\mathbb{X}_2))k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_1) & diag(g(\mathbb{X}_2))k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_2)diag(g(\mathbb{X}_2))+k_{\theta_c'}(\mathbb{X}_2,\mathbb{X}_2)\end{pmatrix}\\ \times\begin{pmatrix}\mathbb{Y}_1-m_1'\mathbb{1}_{n_1}\\ \mathbb{Y}_2-m_1'g(\mathbb{X}_2)-m_c'\mathbf{1}_{n_2}\end{pmatrix}\end{array}\right]\\
\widetilde{\Sigma}_{\eta'} &=& k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_2)\\
&& -\left[\begin{array}{l}\left(k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_1)\quad k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_2)diag(g(\mathbb{X}_2))\right)\\ \times\begin{pmatrix}k_{\theta_1'}(\mathbb{X}_1,\mathbb{X}_1) & k_{\theta_1'}(\mathbb{X}_1,\mathbb{X}_2)diag(g(\mathbb{X}_2))\\ diag(g(\mathbb{X}_2))k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_1) & diag(g(\mathbb{X}_2))k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_2)diag(g(\mathbb{X}_2))+k_{\theta_c'}(\mathbb{X}_2,\mathbb{X}_2)\end{pmatrix}\\ \times\begin{pmatrix}k_{\theta_1'}(\mathbb{X}_1,\mathbb{X}_2)\\ diag(g(\mathbb{X}_2)k_{\theta_1'}(\mathbb{X}_2,\mathbb{X}_2)\end{pmatrix}\end{array}\right]
\end{array}\right.$$

Proof *The proof of this proposition, given in [Zertuche, 2015], is recopied in appendix 9.2.5.*■

Multifidelity with different input spaces for each level This paragraph describes a multifidelity model whose levels of fidelity are not defined in the same input space. Let $\mathcal{X}_1$ denote the input space of f_1, and $\mathcal{X}_2$ the input space of f_2. A multi-fidelity model using deep Gaussian processes has been proposed in [Hebbal et al., 2021]. The output is modelled by

$$Y(x)=\left\{\begin{array}{ll} Y_2(Y_1(x)), & \text{if } x\in\mathcal{X}_1\\ Y_2(Y_1(\Psi_{2\to1}(x))), & \text{if } x\in\mathcal{X}_2\end{array}\right.$$

$Y_1 : \mathcal{X}_1 \to \mathbb{R}$ is a Gaussian process representing the low fidelity level. $Y_2 : \mathcal{X}_2 \to \mathbb{R}$ is a Gaussian process representing the high fidelity level. $\Psi_{2\to1} : \mathcal{X}_2 \to \mathcal{X}_1$ is a multi-output Gaussian process acting as a mapping between the two input spaces. This approach is very complex to implement as it requires a heavy computational machinery.

2.1.4 Sobol index

Sensitivity analysis (see [Da Veiga et al., 2021]) aims at classifying variables by influence on the output. To do so, a well-known criterion is the Sobol index. If $f(x_1,\cdots,x_d)$ is the output, the Sobol index of the variable x_i is defined by:

$$S_i=\frac{Var\left(\mathbb{E}\left[f(X_1,\cdots,X_d)\mid X_i\right]\right)}{Var(f(X_1,\cdots,X_d))},$$

where $X_1, \cdots, X_d$ are independent random variables of uniform distribution in $[0,1]$. The Sobol index S_i computes the proportion of the output variance explained by the input X_i. This index has another formula which is easier to implement numerically with a Monte-Carlo scheme.

Proposition 6 *The Sobol index S_i can be rewritten:*

$$S_i = \frac{Cov(f(X), f(\widetilde{X}))}{Var(f(X))},$$

where $X = (X_1, \cdots, X_{i-1}, X_i, X_{i+1}, \cdots, X_d)$ and $\widetilde{X} = (\widetilde{X}_1, \cdots, \widetilde{X}_{i-1}, X_i, \widetilde{X}_{i+1}, \cdots, \widetilde{X}_d)$. The variable $\widetilde{X}_j$ is independent of same law than X_j.

Proof *The proof of this proposition is given in [Le Gratiet, 2013a] and recopied in appendix 9.2.6 for the reader.* ∎

2.1.5 Design of Experiments

DoE's must be generated to learn the metamodel and to evaluate it. To evaluate the accuracy of the metamodel on a test sample $(\mathbb{X}_{test}, \boldsymbol{y_{test}}) = \left(x_{test}^{(i)}, y_{test}^{(i)}\right)_{i=1}^{n_{test}}$, a usual criterion to use is the RMSE (Root Mean Square Error) defined by:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n_{test}} (y_{test}^{(i)} - \hat{y}(x_{test}^{(i)}))^2}{n_{test}}} \tag{2.2}$$

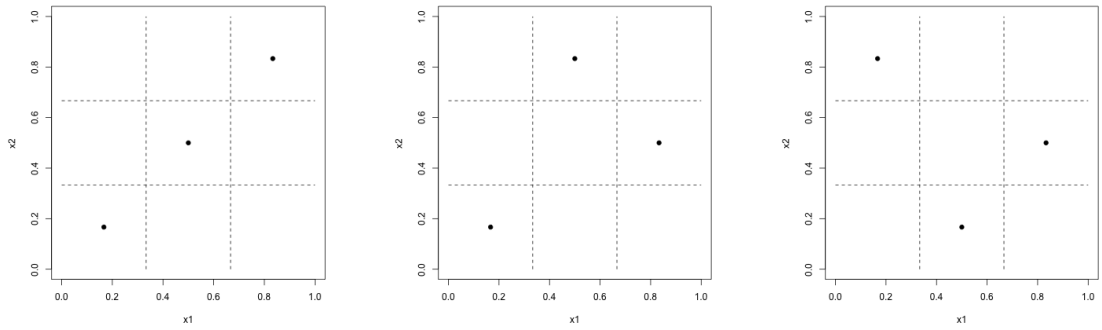
The initial sample on which is built the kriging model must be space filling in order to optimize the learning of the response surface. Different types of DoE's (from [Santner et al., 2003b]) are presented in what follows, with constraints and criteria to optimize.

Latin Hupercube Sampling A very commonly used type of design is the LHS (latin hypercube sample). The purpose of such a design is that the projections of its points in every direction are sufficiently spaced and space filling in the corresponding direction. Basically an LHS design is of the form

$$\mathbb{X} = \begin{pmatrix} \pi_1(u_1) & & \pi_d(u_1) \\ | & \cdots & | \\ \pi_1(u_n) & & \pi_d(u_n) \end{pmatrix},$$

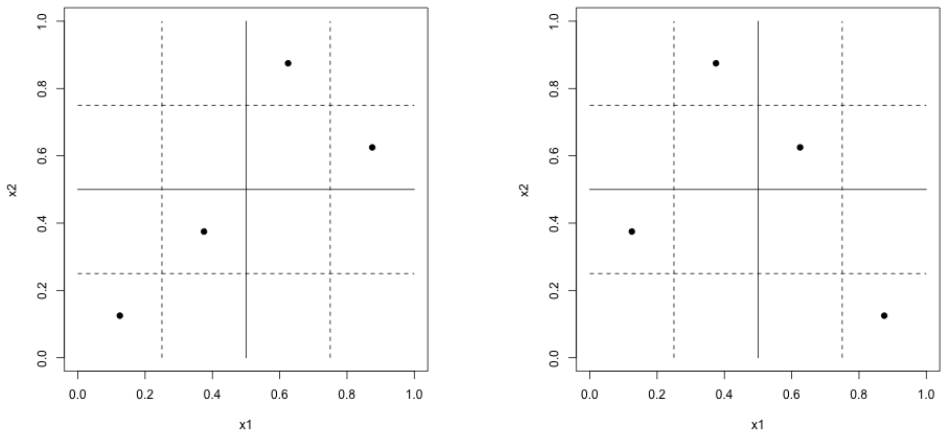
where $\boldsymbol{u} = (u_1, \cdots, u_n)$ is an equispaced discretization of $[0,1]$ and $\pi_1, \cdots, \pi_d$ are permutations on $\boldsymbol{u}$. For example, based on the following discretization $\boldsymbol{u} = \left\{u_1 = \frac{1}{6}, u_2 = \frac{1}{2}, u_3 = \frac{5}{6}\right\}$, the following LHS can be built on $\mathcal{X} = [0,1]^2$:

$$\mathbb{X}_1 = \begin{pmatrix} u_1 & u_1 \\ u_2 & u_2 \\ u_3 & u_3 \end{pmatrix}, \qquad \mathbb{X}_2 = \begin{pmatrix} u_1 & u_1 \\ u_2 & u_3 \\ u_3 & u_2 \end{pmatrix}, \qquad \mathbb{X}_3 = \begin{pmatrix} u_1 & u_3 \\ u_2 & u_1 \\ u_3 & u_2 \end{pmatrix}.$$



All LHS are not necessarily space filling. The choice of the permutations π_i can be restricted to have better space-filling properties

Orthogonal Latin Hypercube (OLH) Sampling An OLH is an LHS which is subject to constraint. The input space is divided in subspaces of same size (the 4 corners in the 2D example below). The points of the OLH must be in the same proportion in all the subspaces.



LHS **OLH**

Those kind of samples are difficult to built. A simpler way to build a space-filling LHS is to choose the permutations π_i by optimizing an additional space filling criterion. The optimization is usually done by simulated annealing.

Space filling criteria

- Maximin: the goal is to maximize the minimal distance between two points of the design

$$\max_{\mathbb{X}} \min_{x,t \in \mathbb{X}} \|x - t\|_2$$

- Generalized maximin: this criterion, which has to be minimized, takes into account the

distances between all pairs of points in the design:

$$\min_{\mathbb{X}} \left(\sum_{x,t \in \mathbb{X}} \left(\frac{1}{\|x-t\|_2} \right)^p \right)^{\frac{1}{p}}.$$

It is reduced to the maximin criterion when $p \to +\infty$.

- Minimax: the goal is to minimize the biggest distance between a point of $\mathbb{X}$ and a point of $\mathcal{X}$:

$$\min_{\mathbb{X}} \max_{x \in \mathbb{X}, t \in \mathcal{X}} \|x-t\|_2$$

- Discrepancy: this criterion, which has to be minimized, measures the uniformity of the distribution of the points of $\mathbb{X}$:

$$\min_{\mathbb{X}} \sup_{J \in \mathcal{J}} \left| \frac{card(\mathbb{X} \cap J)}{n} - \lambda(J) \right|$$

where $\mathcal{J}$ is the set of all hyperrectangles included in $\mathcal{X}$, and λ is the Lebesgues measure.

The discrepancy and minimax criteria are very expensive to compute for a given sample so their optimization is too complex. In practice, the maximin and generalized maximin are used. Alone, they tend to select points at the border of $\mathcal{X}$ especially when the dimension of $\mathcal{X}$ is high (because of the curse of dimensionality), but combined with the latin hypercube constraint, they give suitable designs.

After having trained a kriging model on an initial sample, new points can be added to improve the accuracy of the metamodel. There exists methods to add points relevant to solve problems of optimization [Jones et al., 1998] or inversion [Picheny et al., 2010]. However, here are only presented enrichment methods to improve the accuracy of the metamodel in the whole input space. The procedure consists in adding one or several points optimal for some criterion.

MSE (Mean Squared Error) The additional point maximizes the variance of prediction (defined in equation (2.1)):

$$\max_{x \in \mathcal{X}} \hat{v}(x)$$

It is a space filling criterion. As the variance of prediction increases as the point is far from the DoE, this procedure will add points that will complete the wholes in the DoE. The problem of this method is that the points selected to enrich the sample are often located in the corners of the input space. In the paper [Henner et al., 2019] (see appendix 9.5.1), a methodology is proposed, that selects points optimal for the MSE criterion in a restricted zone around the center point (the input space is $[-1,1]^d$ and the restricted zone is $[-0.5,0.5]^d$).

IMSE (Integrated Mean Squared Error) The point chosen is the one which, when added, minimizes the integrated variance

$$\min_{x} \int_{\mathcal{X}} \hat{v}_{\cup\{x\}}(t) \mathrm{d}t$$

with $\hat{v}_{\cup\{x\}}(t)$ the variance of prediction of the kriging metamodel enriched with the point x.

LOO-CV The two previous criteria are only spatial criteria (favouring points far from the training sample) as they are based on the prediction variance. An interesting idea would be to have a criterion that takes into account both the space and the output evolution. [Le Gratiet, 2013b] suggests an enrichment procedure for the DoE, based on the maximization of the following criterion:

$$\max_{x\in\mathcal{X}} \ \hat{v}(x)\cdot\left[1+\sum_{i=1}^{n}\frac{(y_i-\hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})}\mathbf{1}_{x\in V_i}\right]$$

This criterion is the product of two criteria. The first factor is the prediction variance $\hat{v}(x)$. The second factor is associated to the output evolution, it favors points in the zones where the output behavior is not correctly predicted by the metamodel. y_i is the observation of the output at the training point $x^{(i)}$. $\hat{y}_{-i}$ and $\hat{v}_{-i}$ are respectively the mean and variance of prediction of the kriging metamodel trained on the sample without this point $\mathbb{X}\backslash\{x^{(i)}\}$. V_i is the Voronoï cell corresponding to $x^{(i)}$, i.e. the points that are closer to $x^{(i)}$ than to the other training points. At a given point $x\in\mathcal{X}$, the criterion is equal to $\hat{v}(x)\left(1+\frac{(y_i-\hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})}\right)$ with i the index of the training point $x^{(i)}$ which is the closest to x. This criterion can be seen as an adjusted kriging variance which better estimates the prediction error of the kriging model. Indeed, in the Voronoï cells where the variance supposedly underestimates the prediction error, that means when $(y_i-\hat{y}_{-i}(x^{(i)}))^2 >> \hat{v}_{-i}(x^{(i)})$, the term $\left(1+\frac{(y_i-\hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})}\right)$ automatically increases the value of the criterion, which as a result is superior to the initial variance. Conversely, in the zones where the prediction error is small: $(y_i-\hat{y}_{-i}(x^{(i)}))^2 << \hat{v}_{-i}(x^{(i)})$, the criterion is close to the initial variance $\hat{v}(x)$.

2.2 kriging under constraints : Gaussian process null on any subset

The goal of this section is to review the literature handling directly or indirectly the definition of a Gaussian process null on a part of the input space. Let Z denote such a Gaussian process, defined on $\mathcal{X}_Z$ and null on a subset $\mathcal{D}\subset\mathcal{X}_Z$:

$$Z(x)=0,\ \forall x\in\mathcal{D}\ \Leftrightarrow\ Z(\mathcal{D})=0. \tag{2.3}$$

Three methods of the literature to impose that constraint (or similar ones) are presented in this section. Two methods that are applied for more general constraints (monotonicity, convexicity, boundedness) are described and a method directly dedicated to the nullity constraint is presented.

2.2.1 Imposing the constraint a posteriori

The purpose of the kriging metamodel is the prediction, so the implementation of $\mathbb{E}\left[Z\mid Y=\boldsymbol{y}\right]$ with $Y=\boldsymbol{y}$ some training data. The two methods of the literature described in this subsection work at, amongst others, computing a similar formula, which is of the form $\mathbb{E}\left[W(x)\mid a\leqslant W(\mathcal{D})\leqslant b\right]$, i.e. the expectation of the process W conditioned to be bounded on $\mathcal{D}$.

Discretizing the nullity domain In the paper [Da Veiga and Marrel, 2012], this expectation is first approximated by discretizing $\mathcal{D}$ in $\mathbb{D}$ (where points in $\mathbb{D}$ are choosen uniformly within $\mathcal{D}$). The goal becomes to compute the expectation $\mathbb{E}\left[W(x) \mid a \leqslant W(\mathbb{D}) \leqslant b\right]$. This is the expectation of a truncated normal distribution, whose formula is known analytically in its integral form. The paper then suggests to compute the integrals with adapted numerical approximations. Here, as the inequality constraints are equalities, the expectation is directly computable with the usual formula (see proposition 1).

The drawback of this method is the approximation of $\mathcal{D}$ in $\mathbb{D}$ which does not able to strictly verify the nullity constraint everywhere in $\mathcal{D}$.

Using a finite dimensional GP [Maatouk and Bay, 2017], [López-Lopera et al., 2017], and [Bachoc et al., 2020] tackle the computation of the same expectation $\mathbb{E}\left[W(x) \mid a \leqslant W(\mathcal{D}) \leqslant b\right]$, but this time the inequality constraint must be verified in the whole input space ($\mathcal{D} = \mathcal{X}_W$). The process $W(x)$ is approximated in a finite dimensional way by $\hat{W}(x)$. The method is generalized for any dimension of the input space but is here recalled in 1D for the sake of understanding:

$$\hat{W}(x) = \sum_{j=0}^{m} \xi_j h_j(x),$$

where $\xi = (\xi_0, \ldots, \xi_m)$ is a centered Gaussian vector of covariance matrix denoted by $\Gamma = (k(x^{(i)}, x^{(j)}))_{i,j}$, with k the kernel of W. $h = (h_0, \ldots, h_m)$ is a set of basis functions defined by :

$$h_j(x) = \left(1 - \left|\frac{x - x^{(j)}}{\Delta_m}\right|\right) \mathbf{1}_{\left|\frac{x-x^{(j)}}{\Delta_m}\right| \leqslant 1}$$

with $x^{(j)} = j\Delta_m$ ($j \in [\![0, m]\!]$) a discretization of $[0, 1]$ and $\Delta_m = \frac{1}{m}$ the discretization interval. The expectation $\mathbb{E}\left[\hat{W}(x) \mid a \leqslant \hat{W}(\mathcal{X}) \leqslant b\right]$ is computed by a Monte-Carlo strategy where the paths of $\left[\hat{W}(x) \mid a \leqslant \hat{W}(\mathcal{X}) \leqslant b\right]$ are generated by rejection sampling. It is made possible as it is reduced to simulating the vector $(\xi_0, \cdots, \xi_m)$ with the constraints $a < \xi_i < b$.

The first drawback of this method is that it is expensive in high dimension. Some work has been done in [Bachoc et al., 2020] to apply it at a reduced cost in high dimension but only works if there is a small amount of influential inputs. The second drawback is that the constraint is applied to the whole domain $\mathcal{X}_W$, and its adaptation to a subset $\mathcal{D}$ does not seem straightforward.

2.2.2 Imposing the constraint a priori with a conditional GP

[Gauthier, 2011] directly suggests a candidate for the process Z null on $\mathcal{D}$. This process is presented in definition 6.

Definition 6 *Let $\widetilde{Z} : \Omega \times \mathcal{X}_Z \to \mathbb{R}$ be a centered Gaussian process of law: $\widetilde{Z} \sim \mathcal{GP}(0, k(x, x'))$, with k a positive definite kernel. The Gaussian process defined in [Gauthier, 2011] is equal to:*

$$Z(x) = \widetilde{Z}(x) - \mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(t),\ \forall t \in \mathcal{D}\right],\ \forall x \in \mathcal{X}_Z,$$

where the expectation part is the orthogonal projection of $\widetilde{Z}(x)$ *on the Gaussian space generated by* $\left\{\widetilde{Z}(t),\ t \in \mathcal{D}\right\}$.

Formula of the expectation part for any $\mathcal{D}$

Proposition 7 *$\mathcal{D}$ is supposed to be indexed by a set $\mathcal{S}$: $\mathcal{D} = \{x_s, s \in \mathcal{S}\}$. Let ν be σ-finite measure on $\mathcal{S}$.*

- *The expectation part* $\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(t),\ \forall t \in \mathcal{D}\right]$ *is given by:*

$$\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(s),\ \forall t \in \mathcal{D}\right] = \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}(x_s) \mathrm{d}\nu(s),$$

with $\phi_n : \mathcal{X} \to \mathbb{R}$ *suh that:*

$$\phi_n(x) = \frac{1}{\lambda_n} \int_{\mathcal{S}} k(x, x_s) \widetilde{\phi}_n(s) \mathrm{d}\nu(s),$$

and $(\lambda_n, \widetilde{\phi}_n)$ *solutions of the following eigen problem:*

$$\int_{\mathcal{S}} k(s,u) \widetilde{\phi}_n(u) \mathrm{d}\nu(u) = \lambda_n \widetilde{\phi}_n(s),\ \forall s \in \mathcal{S}.$$

The $\widetilde{\phi}_n : \mathcal{S} \to \mathbb{R}$ *are such that:*

$$\int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{\phi}_m(s) \mathrm{d}\nu(s) = \delta_{nm},\ \forall n, m \in \mathbb{N}\backslash\{0\},$$

with δ_{nm} is the Kronecker symbol ($\delta_{nn} = 1$ and $\delta_{nm} = 0$ if $n \neq m$).

- *The expectation* $\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(t),\ \forall t \in \mathcal{D}\right]$ *is a centered Gaussian process of covariance kernel:*

$$\kappa(x,t) = \sum_{n=1}^{+\infty} \lambda_n \phi_n(x) \phi_n(t).$$

Proof *See [Gauthier and Bay, 2012b]. An alternative proof using the Karhunen-Loeve decomposition is detailed in appendix 9.2.7.* ∎

Formula for finite $\mathcal{D}$

Proposition 8 *If $\mathcal{D} = \mathbb{D}$ is a finite subset of $\mathcal{X}$, then, the expectation part of the process Z coincides with the classical formula of the conditional expectation (see proposition 1):*

$$\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathbb{D})\right] = k(x, \mathbb{D}) k(\mathbb{D}, \mathbb{D})^{-1} \widetilde{Z}(\mathbb{D}).$$

It is a centered Gaussian process of covariance kernel given by:

$$\kappa(x,t) = k(x, \mathbb{D}) k(\mathbb{D}, \mathbb{D})^{-1} k(\mathbb{D}, t)$$

Thus, the process Z coincides with the formula of proposition 2:

$$Z(x) = \widetilde{Z}(x) - k(x, \mathbb{D}) k(\mathbb{D}, \mathbb{D})^{-1} \widetilde{Z}(\mathbb{D}) = \left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathbb{D}) = 0\right].$$

Proof *The proof of this proposition, given in [Gauthier and Bay, 2012a], is recalled in appendix 9.2.8.* ∎

In fact, the process $\mathbb{E}\left[\widetilde{Z} \mid \widetilde{Z}(\mathcal{D})\right]$ (resp. $\widetilde{Z} - \mathbb{E}\left[\widetilde{Z} \mid \widetilde{Z}(\mathcal{D})\right]$), whose formula is given in proposition 7 (resp. definition 6) for any $\mathcal{D}$, is the generalization of the conditional expectation given in proposition 1 (resp. the conditional process $\left[\widetilde{Z} \mid \widetilde{Z}(\mathcal{D}) = 0\right]$ given in proposition 2) which is defined for finite $\mathcal{D}$.

Chapter 3

Probabilistic model

This chapter has two objectives. Firstly, the goal is to suggest a sequential model based on Gaussian processes (called seqGPR) to take into account the results of numerical simulations obtained in spaces of increasing dimension. This model, inspired from the autoregressive multifidelity metamodel of [Kennedy and O'Hagan, 2000], involves processes null on a continuum of points. The second goal is to propose numerically tractable kernels for that kind of processes. For that purpose, the work of [Gauthier, 2011] is used and extended and other kernels are defined.

Section 3.1 defines the probabilistic model on which is based the seqGPR metamodel. Three candidates for the correction processes involved in the model are detailed in section 3.2. The candidates are compared in terms of path shapes and parameter influence in section 3.3. Section 3.4 gives the likelihood formulae for each candidate and compares their parameter estimation by maximum likelihood.

3.1 Model

In this section, a formal probabilistic model of the problem is proposed and the notations that will be used in the rest of the thesis are defined.

3.1.1 General formalism

Let $f : [0,1]^{d_1+\ldots+d_N} \to \mathbb{R}$ be an output of an expensive simulation code depending on $d_1 + \ldots + d_N$ variables. A response surface of f has to be built taking into account that the $N-1$ first steps of the study (of increasing dimensions) have already been completed. Each of these steps is a focus on the relation between the output and a subset of free variables. The other variables are temporarily fixed but then progressively released in the further steps. This thesis deals with the handling of the last step of the study, denoted by N, where all inputs are free.
Let I_n denote the index set of the variables that are released at step $n \in [\![1, N]\!]$ and $d_n = card(I_n)$. For every index set I, let x_I denote the associated subvector of x. The following framework, based on Gaussian process regression, is introduced to model the sequential study:

- At step 1, a first series of computer code evaluations is run. The set x_{I_1} of the first d_1 components of x, are free in $[0,1]^{d_1}$. Different values are explored. They are stored in DoE $\mathbb{X}_1 \subset [0,1]^{d_1}$. The other components are fixed to preset values $\grave{x}_{I_2 \cup \ldots \cup I_N}$ (entirely determined by x_{I_1}). Let f_1 denote the corresponding restriction of f on the subspace

$[0,1]^{d_1}$. The function f_1 is assumed to be the realization of a stationary Gaussian process $Y_1 = m + Z_1$ of mean $m \in \mathbb{R}$ and with $Z_1 : \Omega \times [0,1]^{d_1} \to \mathbb{R}$ a centered Gaussian Process of covariance kernel $\sigma_1^2 \rho_1$. Let $\boldsymbol{y_1} = f_1(\mathbb{X}_1)$ represent the vector of observations of the output on DoE $\mathbb{X}_1$.

- At step 2, a second range of simulations is then launched on a subspace of higher dimension $[0,1]^{d_1+d_2}$. The variables x_{I_2}, fixed at step 1, are released. Different values of $x_{I_1 \cup I_2}$, stored in DoE $\mathbb{X}_2$, are explored. The other variables $x_{I_3 \cup \cdots \cup I_N}$ are fixed to the new set of values $\grave{x}_{I_3 \cup \cdots \cup I_N} \in [0,1]^{d_3+\cdots+d_N}$. Let f_2 be the corresponding restriction of f on the subspace $[0,1]^{d_1+d_2}$.

  The function f_2 is assumed to be the realization of a Gaussian process $Y_2 : \Omega \times [0,1]^{d_1+d_2} \to \mathbb{R}$, which is the sum of the process at step 1 Y_1, and a correction term Z_2, which represents the additional information provided by the released variables. As f_1 and f_2 are restrictions of the same function f, Y_1 and Y_2 have to coincide on the subspace defined at step 1. This results in the following definition of Y_2:

$$Y_2(x_{I_1}, x_{I_2}) = Y_1(x_{I_1}) + Z_2(x_{I_1}, x_{I_2}), \quad \forall (x_{I_1}, x_{I_2}) \in [0,1]^{d_1+d_2},$$

  with Z_2 a centered Gaussian process independent from Z_1, of covariance kernel $\sigma_2^2 \rho_2$ such that for all x_{I_1} in $[0,1]^{d_1}$, $Z_2(x_{I_1}, \grave{x}_{I_2}) = 0$. Let $\boldsymbol{y_2} = f_2(\mathbb{X}_2)$ denote the vector of observations for this step.

- In the same way as previous steps, at step $n \in [\![3, N]\!]$, a DoE $\mathbb{X}_n$ is created in the higher space $[0,1]^{d_1+\cdots+d_n}$. The variables $x_{I_1 \cup \cdots \cup I_{n-1}}$ and x_{I_n} are free and the others $x_{I_{n+1} \cup \cdots \cup I_N}$ are set to $\grave{x}_{I_{n+1} \cup \cdots \cup I_N} \in [0,1]^{d_{n+1}+\cdots+d_N}$. The corresponding restriction of f on this subspace, f_n, is evaluated on $\mathbb{X}_n$. The values are stored in $\boldsymbol{y_n} = f_n(\mathbb{X}_n)$. The function f_n is modeled as the realization of a Gaussian process $Y_n : \Omega \times [0,1]^{d_1+\cdots+d_n} \to \mathbb{R}$. Following the same arguments as in step 2, Y_n is defined as (for all $(x_{I_1 \cup \cdots I_{n-1}}, x_{I_n})$ in $[0,1]^{d_1+\cdots+d_n}$):

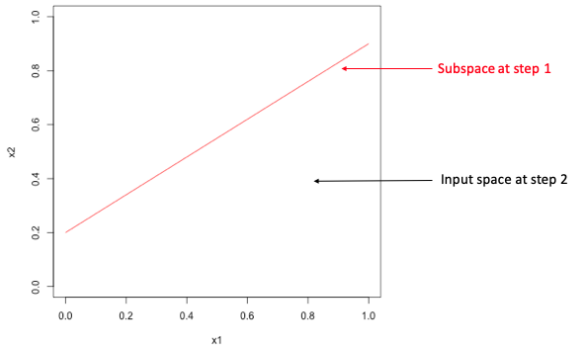
$$Y_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}) = Y_{n-1}(x_{I_1 \cup \cdots \cup I_{n-1}}) + Z_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}),$$

  with Z_n a centered Gaussian process independent from $(Z_1, \cdots, Z_{n-1})$ of covariance kernel $\sigma_n^2 \rho_n$ and such that $Z_n(x_{I_1 \cup \cdots \cup I_{n-1}}, \grave{x}_{I_n}) = 0, \forall x_{I_1 \cup \cdots \cup I_{n-1}} \in [0,1]^{d_1+\cdots+d_{n-1}}$.

3.1.2 Examples

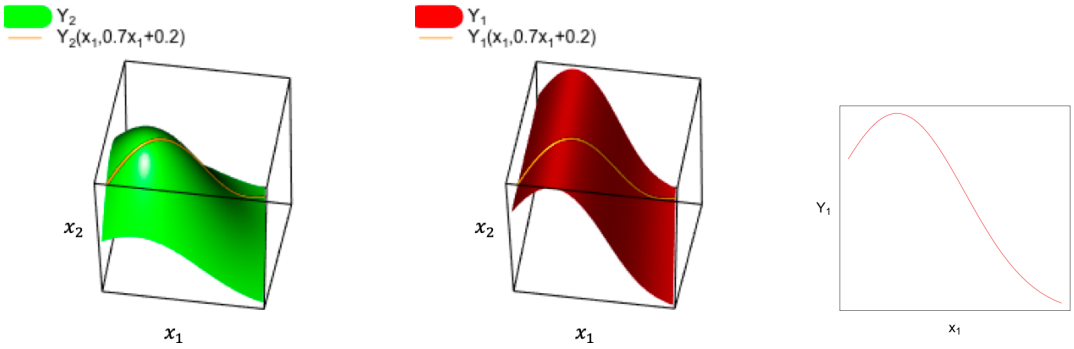
Example in 2D This example will follow the course of the thesis to illustrate the method in a simple case. Let $f(x_1, x_2)$ be a function of two variables to approximate by a metamodel. A two-step study is carried out:

- At step 1, the input x_2 is fixed to $0.7x_1 + 0.2$. The subspace of the input space considered at this step is shown in figure below.



Here, this subspace is not indexed by the curvilinear abscissa ($a_c = \sqrt{x_1^2 + (0.7x_1 + 0.2)^2}$) but simply by x_1: $\{(x_1, 0.7x_1 + 0.2), x_1 \in [0,1]\}$. The function to approximate is $f_1(x_1) = f(x_1, 0.7x_1 + 0.2)$. This restriction is modeled by a Gaussian process $Y_1(x_1) = m + Z_1(x_1)$ with m a scalar equal to the mean of Y_1, and Z_1 a centered Gaussian process of covariance kernel $k_1(x_1, t_1)$. Thus, Y_1 is a function of x_1 (and not of the curvilinear abscissa a_c). A DoE is generated $\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}$ containing values of x_1. For these points, the values of x_2 are implicitely fixed to $0.7x_1 + 0.2$. The values of f_1 at the points of $\mathbb{X}_1$ are known, equal to $\boldsymbol{y_1} = \begin{pmatrix} f_1(x_1^{(1)}) \\ \vdots \\ f_1(x_1^{(n_1)}) \end{pmatrix}$.

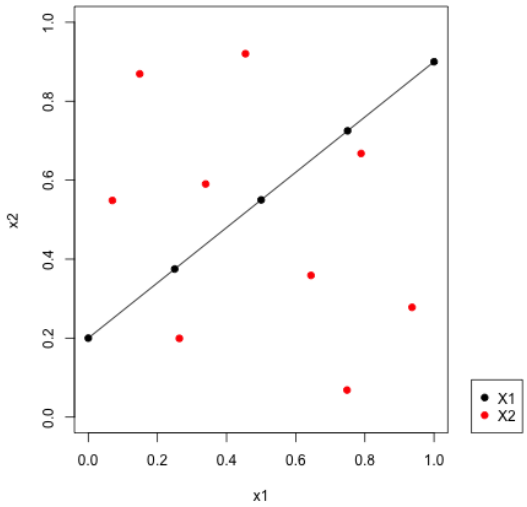
- At step 2, the entire input space $[0,1]^2$ is considered. The function to approximate is now $f_2(x_1, x_2) = f(x_1, x_2)$. It is modeled by a Gaussian process $Y_2(x_1, x_2)$ linked to $Y_1(x_1)$ by the formula $Y_2(x_1, x_2) = Y_1(x_1) + Z_2(x_1, x_2)$. This formula looks like the multifidelity metamodel, but the difference is that Y_1 and Y_2 model the same level of fidelity. Thus, they must coincide on the subspace $x_2 = 0.7x_1 + 0.2$. This coincidence, shown in the figure below, implies that $Z_2(x_1, 0.7x_1 + 0.2) = 0$. Y_1 can be seen as a function of (x_1, x_2) constant in x_2 (see the middle panel). On the left and middle panels, Y_1 and Y_2 coincide on the orange line corresponding to the subspace $x_2 = 0.7_1 + 0.2$. The right panel shows Y_1 as a function of x_1.



Visualization of Y_2 **Y_1 as a function of (x_1, x_2)** **Y_1 as a function of x_1**

A DoE is generated $\mathbb{X}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} \\ \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} \end{pmatrix}$ containing values of (x_1, x_2). The

values of f_2 at the points of $\mathbb{X}_2$ are known, equal to $\boldsymbol{y_2} = \begin{pmatrix} f_2(x_1^{(n_1+1)}, x_2^{(n_1+1)}) \\ \vdots \\ f_2(x_1^{(n_1+n_2)}, x_2^{(n_1+n_2)}) \end{pmatrix}$. An example of the DoE's $\mathbb{X}_1$ and $\mathbb{X}_2$ is shown in the figure below:



Example in 4D This example will follow the course of the thesis to illustrate the general definitions in a particular case, which tries to encompass the problem in its generality: variables fixed at constants or deterministic functions, variables released one by one or several at a time, and a study composed of more than two steps.

In this example, $N = 3$, $I_1 = 1$, $I_2 = \{2, 3\}$, $I_3 = 4$, $(\grave{x}_2, \grave{x}_3) = (0.4, 0.5)$, $\grave{x}_4 = \frac{x_1+x_2+x_3}{4}$. The output considered is a function of 4 inputs $f(x_1, x_2, x_3, x_4)$. The study is composed of 3 steps.

- At step 1, x_1 is released and (x_2, x_3, x_4) are fixed. (x_2, x_3) are equal to $(0.4, 0.5)$. x_4 is equal to $\frac{x_1+x_2+x_3}{4} = \frac{x_1+0.9}{4}$. So the restriction considered is $f_1(x_1) = f\left(x_1, 0.4, 0.5, \frac{x_1+0.9}{4}\right)$. f_1 is assumed to be the realization of $Y_1(x_1) = m + Z_1(x_1)$, with Z_1 a centered Gaussian process of covariance kernel $\sigma_1^2 \rho_1(x_1, t_1)$. The DoE $\mathbb{X}_1$ is of the form:

$$\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}.$$

- At step 2, (x_2, x_3) are released and x_4 is still fixed. x_4 is equal to $\frac{x_1+x_2+x_3}{4}$. The restriction considered is $f_2(x_1, x_2, x_3) = f\left(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}\right)$. f_2 is assumed to be the realization of $Y_2(x_1, x_2, x_3) = Y_1(x_1) + Z_2(x_1, x_2, x_3)$, with Z_2 a centered Gaussian process of covariance kernel $\sigma_2^2 \rho_2\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right)$ such that $Z_2(x_1, 0.4, 0.5) = 0$, for all x_1 in $[0, 1]$. This property of Z_2 enables the following equality $Y_2(x_1, 0.4, 0.5) = Y_1(x_1)$, for all x_1 in $[0, 1]$. The DoE $\mathbb{X}_2$ is of the form:

$$\mathbb{X}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ & \vdots & \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} & x_3^{(n_1+n_2)} \end{pmatrix}.$$

- At step 3, x_4 is released. All 4 inputs are free so the whole output f is considered. It is assumed to be the realization of $Y_3(x_1, x_2, x_3, x_4) = Y_2(x_1, x_2, x_3) + Z_3(x_1, x_2, x_3, x_4)$, with Z_3 a centered Gaussian process of covariance kernel $\sigma_3^2 \rho_3 ((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4))$ such that $Z_3(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}) = 0$, to enable $Y_3(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}) = Y_2(x_1, x_2, x_3)$, for all (x_1, x_2, x_3) in $[0, 1]^3$. The DoE $\mathbb{X}_3$ is of the form:

$$\mathbb{X}_3 = \begin{pmatrix} x_1^{(n_1+n_2+1)} & x_2^{(n_1+n_2+1)} & x_3^{(n_1+n_2+1)} & x_4^{(n_1+n_2+1)} \\ & \vdots & & \\ x_1^{(n_1+n_2+n_3)} & x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} & x_4^{(n_1+n_2+n_3)} \end{pmatrix}$$

3.1.3 Metamodel seqGPR (Sequential Gaussian process regression)

Finally, the problem is modeled by the following statistical model (for $x_{I_1 \cup \cdots \cup I_n}$ in $[0, 1]^{d_1+\cdots+d_n}$):

$$\begin{cases} Y_1(x_{I_1}) & = & m + Z_1(x_{I_1}), \\ Y_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}) & = & Y_{n-1}(x_{I_1 \cup \cdots \cup I_{n-1}}) + Z_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}), \quad \forall n \in [\![2, N]\!], \end{cases} \tag{3.1}$$

where:

- The processes $(Z_n)_{n=1}^N$ are independent Gaussian processes of law:

$$Z_n \sim \mathcal{GP}(0, \sigma_n^2 \rho_n(x_{I_1 \cup \cdots \cup I_n}, t_{I_1 \cup \cdots \cup I_n})), \quad \forall n \in [\![1, N]\!]. \tag{3.2}$$

- The processes $(Z_n)_{n=2}^N$ verify the following property:

$$Z_n(x_{I_1 \cup \cdots I_{n-1}}, \grave{x}_{I_n}) = 0, \quad \forall n \in [\![2, N]\!], \forall x_{I_1 \cup \cdots I_{n-1}} \in [0, 1]^{d_1+\cdots+d_{n-1}}. \tag{3.3}$$

The same formulae of prediction as for a classic kriging metamodel apply (see subsection 2.1.1 in chapter 2). For every $x \in [0, 1]^d$, the prediction mean $\hat{y}(x)$ and variance $\hat{v}(x)$ are defined by:

$$\begin{cases} \hat{y}(x) & = & \mathbb{E}\left[Y_N(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, \cdots, Y_N(\mathbb{X}_N) = \boldsymbol{y_N}\right], \\ \hat{v}(x) & = & Var\left(Y_N(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, \cdots, Y_N(\mathbb{X}_N) = \boldsymbol{y_N}\right). \end{cases}$$

Remark The independence hypothesis on the $(Z_n)_{1 \leqslant n \leqslant N}$ is added as in the multifidelity model [Kennedy and O'Hagan, 2000], from which this model is inspired. The dependence is privileged between the $(Y_n)_{1 \leqslant n \leqslant N}$ which really model the output. Furthermore, this hypothesis simplifies calculations for the likelihood and so for the parameter estimation, as shown in section 4.1 in chapter 4.

3.2 Candidates for the correction processes

In this section, candidates for the correction processes, that must satisfy the nullity property (3.3), are suggested. More precisely, the goal is to build a process $Z : [0, 1]^{d_J+d_I} \times \Omega \to \mathbb{R}$ null on an infinite continuous set of points :

$$Z(x_J, g(x_J)) = 0 \quad \forall x_J \in [0, 1]^{d_J}, \tag{3.4}$$

with $g : [0,1]^{d_J} \rightarrow [0,1]^{d_I}$ a deterministic function.

Three candidates are suggested for the process Z, all based on a latent Gaussian process $\widetilde{Z} \sim \mathcal{GP}(0, \sigma^2 r((x_J, x_I), (t_J, t_I)))$ of covariance kernel $\sigma^2 r$. The function g is imposed by industrial concerns, however one can discuss about its regularity. If the goal is to have a Z kernel of the same regularity than the $\widetilde{Z}$ kernel, then a reasonable rule may be that g must at least be as regular as the kernel of $\widetilde{Z}$.

The first candidate results from a reduction of the latent process and is called the Red process. The second candidate, called the Psi process, involves a multiplicating term denoted by Ψ, null on the zone of nullity. The third candidate is a tractable version of the process defined by [Gauthier, 2011], which has been recalled in subsection 2.2.2 in chapter 2.

3.2.1 Red (Reduced) process

The transformation leading to the Red process consists in subtracting from $\widetilde{Z}$ its value on the subspace $\{(x_J, g(x_J)), x_J \in [0,1]^{d_J}\}$. Thus, the Red process is equal to:

$$Z^{Red}(x_J, x_I) = \widetilde{Z}(x_J, x_I) - \widetilde{Z}(x_J, g(x_J)).$$

It is a centered Gaussian process of covariance kernel $\sigma^2 \rho^{Red}$ with:

$$\begin{aligned} \rho^{Red}((x_J, x_I), (t_J, t_I)) &= r((x_J, x_I), (t_J, t_I)) - r((x_J, x_I), (t_J, g(t_J))) \\ &\quad - r((x_J, g(x_J)), (t_J, t_I)) + r((x_J, g(x_J)), (t_J, g(t_J))). \end{aligned}$$

As the process used to transform the latent process is constant in x_I, the transformation is global and not just near the nullity zone. The resulting process is entirely disturbed.

3.2.2 Psi process

Another transformation to impose the nullity property while modifying the latent process only near the nullity subspace consists in multiplying the latent process by a function null on the nullity subspace $\{(x_J, g(x_J)), x_J \in [0,1]^{d_J}\}$ and equal to one far from it. The resulting process is called the Psi process:

$$Z^{Psi}(x_J, x_I) = \Psi(x_I - g(x_J))\, \widetilde{Z}(x_J, x_I).$$

with Ψ equal to:

$$\Psi(t) = 1 - \exp\left(-\sum_{i=1}^{d_I} \frac{t_i^2}{2\delta_i^2}\right)$$

The parameter δ_i ($i \in [\![1, d_I]\!]$) controls the speed at which Ψ goes to 0 in the direction t_i. The smaller δ_i is, the steeper is the slope toward 0.

The Psi process is a centered Gaussian process of covariance kernel $\sigma^2 \rho^{Psi}$ with:

$$\rho^{Psi}((x_J, x_I), (t_J, t_I)) = \Psi(x_I - g(x_J))\, \Psi(t_I - g(t_J))\, r((x_J, x_I), (t_J, t_I))$$

As the Psi function used to transform the latent process is almost constant equal to 1 except near the nullity zone where it quickly goes down to 0, the transformation is local, located around the nullity zone. The resulting process is disturbed near the nullity zone, but the disturbation is important and not smooth.

3.2.3 P (Preconditioned) process

An alternative local and smooth transformation of the latent process to impose the nullity property consists in conditioning the latent process to be null on the nullity subspace $\mathcal{D} = \{(x_J, g(x_J)), x_J \in [0,1]^{d_J}\}$. As this nullity subspace is an infinite continuous set of points, the usual conditioning seen in proposition 2 in chapter 2 cannot be used directly. Instead, a generalization of the conditioning defined in [Gauthier and Bay, 2012b] and detailed in subsection 2.2.2 in chapter 2 is used. The resulting process is called the P process:

$$Z^P(x_J, x_I) \;=\; \widetilde{Z}(x_J, x_I) - \mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid \widetilde{Z}(s_J, g(s_J)), \forall s_J \in [0,1]^{d_J}\right]. \tag{3.5}$$

The term $\mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid \widetilde{Z}(s_J, g(s_J)), \forall s_J \in [0,1]^{d_J}\right]$ is the orthogonal projection of $\widetilde{Z}(x_J, x_I)$ in the sub Gaussian Space engendered by the family $\widetilde{Z}(\mathcal{D}) = \left(\widetilde{Z}(s_J, g(s_J))\right)_{s_J \in [0,1]^{d_J}}$. Its general expression is given in subsection 2.2.2 in chapter 2 and is recalled below:

$$\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathcal{D})\right] \;=\; \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}(x_s) \mathrm{d}\nu(s), \tag{3.6}$$

with $\mathcal{D} = \{x_s, s \in \mathcal{S}\}$ the nullity subspace, $\mathcal{S}$ its index set, ν a measure on $\mathcal{S}$, and

$$\phi_n(x) \;=\; \tfrac{1}{\lambda_n} \textstyle\int_{\mathcal{S}} \sigma^2 r(x, x_s) \widetilde{\phi}_n(s) \mathrm{d}\nu(s).$$

$(\lambda_n, \widetilde{\phi}_n)_{n \geqslant 1}$ are solutions of the eigen problem :

$$\int_{\mathcal{S}} \sigma^2 r\left(x_s, x_u\right) \widetilde{\phi}_n(u) \mathrm{d}\nu(u) = \lambda_n \widetilde{\phi}_n(s), \;\; \forall s \in \mathcal{S},$$

such that

$$\int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{\phi}_m(s) \mathrm{d}\nu(s) = \delta_{nm} \;\; \forall n, m \geqslant 1,$$

where δ_{nm} is the Kronecker symbol. It is applied to the case:

$$\begin{cases} \mathcal{S} = [0,1]^{d_J}, \\ \nu \text{ any measure on } [0,1]^{d_J}, \\ x_s = (s_J, g(s_J)). \end{cases}$$

This kernel cannot be used just as it is in practice, because the solutions of the eigenvalue problem are not explicit in general and the sums are infinite. Two ways of computing this kernel are proposed. The first is based on the discretization of the spectral decomposition. The second is based on a wise choice of r (the correlation kernel of the latent process $\widetilde{Z}$) for which an explicit formula is known.

In this paragraph, a first way of computing the kernel is given by an approximation based on the discretization of the spectral decomposition that reduces the functional eigenvalue problem to a finite dimensional one. In that case, the terms from the spectral decomposition vanish and the formula of the resulting approximate kernel only depends on the kernel of the latent process.

Proposition 9 *Discretizing the spectral decomposition of the P process Z by a quadrature method (as in the 2D example in subsection 3.2.4) using weights $(\omega_i)_{i=1}^L$, quadrature points $(s^{(i)})_{i=1}^L$, and the associated discretized nullity subspace $\mathbb{D} = \left\{(s^{(i)}, g(s^{(i)})), i \in [\![1, L]\!]\right\}$, is equivalent to approximating the P process by the process $\widetilde{Z}$ conditioned to be null on the points of $\mathbb{D}$:*

$$Z^{\mathbb{D}} = \left[\widetilde{Z} \mid \widetilde{Z}(\mathbb{D}) = 0\right].$$

It is a centered Gaussian process of covariance kernel $\sigma^2\rho^{\mathbb{D}}$:

$$\rho^{\mathbb{D}}(x,t) = r(x,t) - r(x,\mathbb{D})r(\mathbb{D},\mathbb{D})^{-1}r(\mathbb{D},t) \quad \forall x,t \in [0,1]^{d_J+d_I}.$$

Proof *It is a straightforward application of proposition 8 with $\mathcal{D} = \mathbb{D}$, $\mathcal{S} = \mathbb{S}$, and $\nu = \sum_{i=1}^L \omega_i \delta_{s^{(i)}}$. An alternative proof is suggested in appendix 9.3.1 in the case of a Monte-Carlo method.* ∎

This process does not suit the original expectations as the nullity is not fully respected on the continous set of points. The size of $\mathbb{D}$ must be high to approach correctly the full nullity. It is more and more difficult as the dimension of the input space increases, and it leads to very expensive computations (with inversion of huge matrices). A second way of computing this kernel is proposed. It consists in finding forms of the kernel of $\widetilde{Z}$ that enable the kernel of Z^P to be tractable.

Proposition 10 *Let r_J and r_I be two stationary correlation kernels. If the covariance kernel of $\widetilde{Z}$ is of the form $\sigma^2 r$ with:*

$$r((x_J,x_I),(t_J,t_I)) = r_J(x_J,t_J)r_I(x_I - g(x_J), t_I - g(t_J)),$$

then the P process is a centered Gaussian process equal to :

$$Z^P(x_J,x_I) = \widetilde{Z}(x_J,x_I) - r_I(x_I - g(x_J),0)\widetilde{Z}(x_J,g(x_J)),$$

whose covariance kernel is $\sigma^2\rho^P$ with:

$$\rho^P((x_J,x_I),(t_J,t_I)) = r_J(x_J,t_J)\left[r_I(x_I - g(x_J), t_I - g(t_J)) - r_I(x_I - g(x_J),0)r_I(0, t_I - g(t_J))\right].$$

Proof *See appendix 9.3.2 for the proof of this proposition.* ∎

Corollary 1 *If g is constant equal to $c \in [0,1]^{d_I}$ and if r is a stationary kernel of the form*

$$r((x_J,x_I),(t_J,t_I)) = r_J(x_J,t_J)r_I(x_I,t_I),$$

then the P process is a centered Gaussian process equal to :

$$Z^P(x_J,x_I) = \widetilde{Z}(x_J,x_I) - r_I(x_I,c)\widetilde{Z}(x_J,g(x_J)),$$

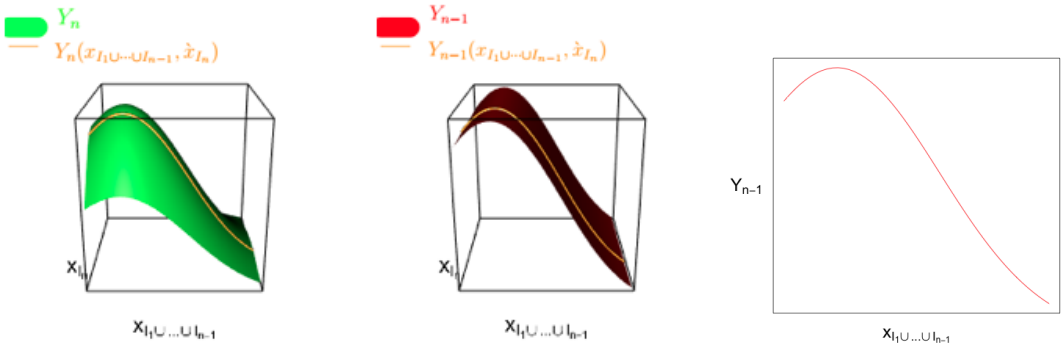
whose covariance kernel is $\sigma^2\rho^P$ with:

$$\rho^P((x_J,x_I),(t_J,t_I)) = r_J(x_J,t_J)\left[r_I(x_I,t_I) - r_I(x_I,c)r_I(c,t_I)\right].$$

Proof *The proof is straightforward, by applying proposition 10 with $g(x_J) = c$ and using the stationarity of r_I : $r_I(x_I - c, t_I - c) = r_I(x_I,t_I)$.* ∎

The model presented in the corollary (where the variables that are released were previously fixed at a constant value) is illustrated in figure 3.1.

The expectation used to transform the latent process is smooth and equal to zero near the nullity zone. The transformation is local, located around the nullity zone. The resulting process is disturbed near the nullity zone, but the disturbation is smooth. In the following, the P processes are assumed to be built as in proposition 10 or corrollary 1.



(a) Visualization of Y_n and Y_{n-1} (b) Isocontours of Y_n (c) Visualization of Y_{n-1}

Figure 3.1: Illustration of the coincidance of Y_n and Y_{n-1} when $\dot{x}_{I_n}$ is constant.

3.2.4 Example in 2D

This subsection describes the three candidates in the context of the 2D example defined in subsection 3.1.2. They are candidates for the correction process Z_2 which verifies the nullity property $Z_2(x_1, g(x_1)) = 0$, where $g(x_1) = 0.7x_1 + 0.2$.

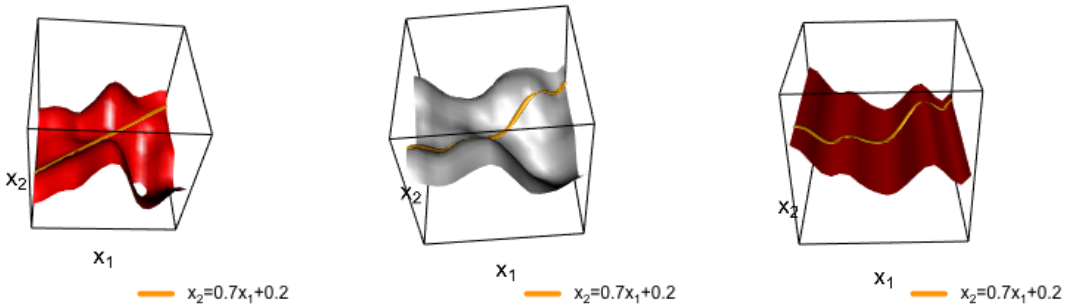
Red process If Z_2 is a Red process, it is defined as:

$$Z_2(x_1, x_2) = \widetilde{Z}_2(x_1, x_2) - \widetilde{Z}_2(x_1, 0.7x_1 + 0.2).$$

This transformation of $\widetilde{Z}_2$ is linear and therefore, the Gaussian law is preserved. The nullity property is easily verified and the kernel can directly be expressed as a linear transformation of the latent kernel so stays computable. The covariance kernel of Z_2 is equal to $\sigma_2^2 \rho_2^{Red}$ with:

$$\begin{aligned} \rho_2^{Red}((x_1, x_2), (t_1, t_2)) &= r_2((x_1, x_2), (t_1, t_2)) + r_2((x_1, 0.7x_1 + 0.2), (t_1, 0.7t_1 + 0.2)) \\ &- r_2((x_1, x_2), (t_1, 0.7t_1 + 0.2)) - r_2((x_1, 0.7x_1 + 0.2), (t_1, t_2)). \end{aligned}$$

The different processes involved are shown in figure below: the Red process in left panel, the latent process on middle panel, and its value on the line $x_2 = 0.7x_1 + 0.2$ (which is a process function of x_1 that can be seen as a process function of (x_1, x_2) constant in x_2) on right panel.

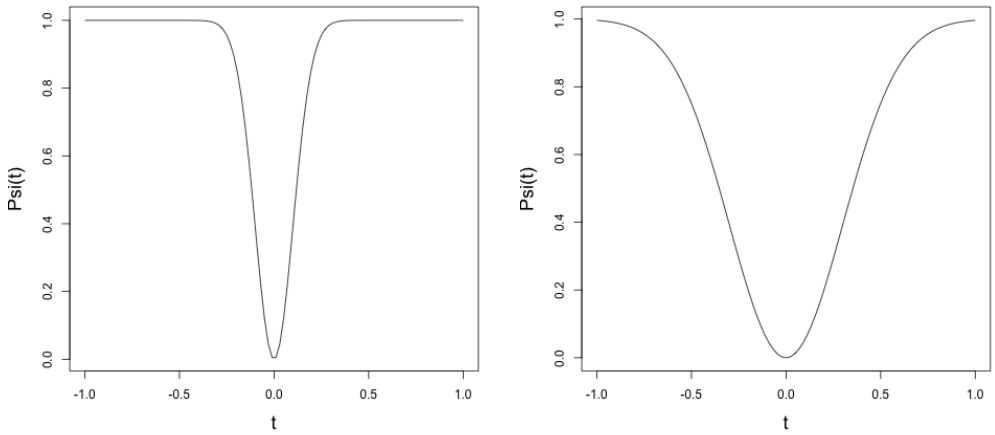


$\boldsymbol{Z_2^{Red}(x_1, x_2)}$ $\boldsymbol{\widetilde{Z}_2(x_1, x_2)}$ $\boldsymbol{\widetilde{Z}_2(x_1, 0.7x_1 + 0.2)}$

Psi process The Psi process is defined as:

$$Z_2(x_1, x_2) = \Psi(x_2 - 0.7x_1 - 0.2)\widetilde{Z}_2(x_1, x_2),$$

with $\Psi(t) = 1 - \exp\left(\frac{t^2}{2\delta^2}\right)$ for all t in $[-1, 1]$. The function Ψ is null at 0 and near 1 far from 0, and the parameter δ controls the speed at which Ψ goes to 0. The smaller δ is, the steeper is the slope toward 0, as illustrated on the figure below.

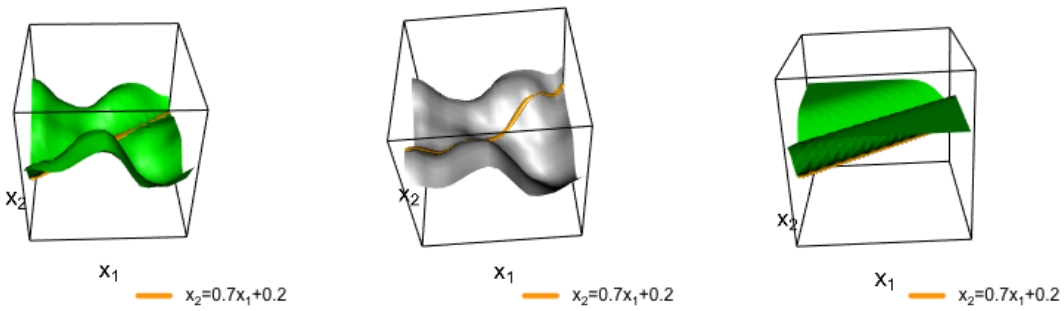


$\boldsymbol{\delta = 0.1}$ $\boldsymbol{\delta = 0.3}$

The covariance kernel of Z_2 is equal to $\sigma_2^2 \rho_2^{Psi}$ with:

$$\rho_2((x_1, x_2), (t_1, t_2)) \quad = \quad \Psi(x_2 - 0.7x_1 - 0.2)\Psi(t_2 - 0.7t_1 - 0.2)r_2((x_1, x_2), (t_1, t_2)).$$

The different processes involved are shown in figure below: the Psi process in left panel, the latent process on middle panel, and the function $\Psi(x_2 - 0.7x_1 - 0.2)$ on right panel.



$\boldsymbol{Z_2^{Psi}(x_1, x_2)}$ $\boldsymbol{\widetilde{Z}_2(x_1, x_2)}$ $\boldsymbol{\Psi(x_2 - 0.7x_1 - 0.2)}$

P process The P process is defined as:

$$Z_2(x_1, x_2) = \widetilde{Z}_2(x_1, x_2) - \mathbb{E}\left[\widetilde{Z}_2(x_1, x_2) \mid \widetilde{Z}_2(t_1, 0.7t_1 + 0.2)\forall t_1 \in [0, 1]\right],$$

where $\mathbb{E}\left[\widetilde{Z}_2(x_1, x_2) \mid \widetilde{Z}_2(t_1, 0.7t_1 + 0.2)\forall t_1 \in [0, 1]\right]$ (denoted by $E(x_1, x_2)$ to simplify notations) is itself a generalization of the conditional expectation, defined as the projection of $\widetilde{Z}_2(x_1, x_2)$ in the subGaussian space generated by $\left(\widetilde{Z}_2(t_1, 0.7t_1 + 0.2)\right)_{t_1 \in [0,1]}$. The formula

of the expectation, taken from the proposition 7 and recalled in the next paragraph, is here adapted to the case $\mathcal{D} = \{(s, 0.7s + 0.2), s \in [0, 1]\}$, $\mathcal{S} = [0, 1]$, and ν any measure on $\mathcal{S}$:

$$E(x_1, x_2) = \sum_{n=1}^{+\infty} \phi_n(x_1, x_2) \int_0^1 \widetilde{\phi}_n(s) \widetilde{Z}_2(s, 0.7s + 0.2) \mathrm{d}\nu(s),$$

where:

$$\phi_n(x_1, x_2) = \frac{1}{\lambda_n} \int_0^1 \sigma_2^2 r_2((x_1, x_2), (s, 0.7s + 0.2)) \widetilde{Z}_2(s, 0.7s + 0.2) \mathrm{d}\nu(s),$$

and $(\lambda_n, \widetilde{\phi}_n)_{n=1}^{+\infty}$ are the solutions of the eigenvalue problem:

$$\int_0^1 \sigma_2^2 r_2((s, 0.7s + 0.2), (u, 0.7u + 0.2)) \widetilde{\phi}_n(u) \mathrm{d}\nu(u) = \lambda_n \widetilde{\phi}_n(s).$$

$(\widetilde{\phi}_n)_{n \geqslant 1}$ is an orthonormal basis of $L^2([0, 1], \nu)$:

$$\int_0^1 \widetilde{\phi}_n(s) \widetilde{\phi}_m(s) \mathrm{d}\nu(s) = \delta_{nm}.$$

The first way to compute its kernel is based on the discretization of its spectral decomposition. Discretizing the spectral decomposition of the P process means that the integrals are approximated by quadrature methods, i.e. $\int_0^1 g(s) \mathrm{d}\nu(s) \approx \sum_{i=1}^L \omega_i g(s^{(i)})$ with $(\omega_i)_{i=1}^L$ the weigths and $(s^{(i)})_{i=1}^L$ the quadrature points choosen in $[0, 1]$. For example, in the case of the Monte-Carlo method, $\omega_i = \frac{1}{L}$ and $\mathbb{S} = \left(s^{(i)}\right)_{i=1}^L$ is choosen uniformly in $[0, 1]$. Using a quadrature approximation of the integral, the eigenvalue problem becomes:

$$\sum_{i=1}^L \omega_i \sigma_2^2 r_2((s, 0.7s + 0.2), (s^{(i)}, 0.7s^{(i)} + 0.2)) \widetilde{\phi}(s^{(i)}) \approx \lambda \widetilde{\phi}(s), \ \forall s \in [0, 1].$$

Then, by considering only the values of the eigenfunction $\widetilde{\phi}$ at the quadrature points $(s^{(i)})_{i=1}^L$, the eigenvalue problem is replaced by a finite dimensional one:

$$\sigma_2^2 r_2(\mathbb{D}, \mathbb{D}) W \Phi = \gamma \Phi,$$

where $\mathbb{D} = (s^{(i)}, 0.7s^{(i)} + 0.2)_{i=1}^L$ is the discretization of the nullity subspace $\mathcal{D}$ corresponding to $\mathbb{S}$ (discretization of $\mathcal{S}$), $W = \begin{pmatrix} \omega_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \omega_L \end{pmatrix}$ is the diagonal matrix whose diagonal components are the weights of the quadrature method. According to proposition 7, there exist $(\gamma_n, \Phi_n)_{n=1}^L$ solutions of this finite dimensional eigenvalue problem such that $(\Phi_n)_{n=1}^L$ is an orthonormal basis of $\mathbb{R}^L$ for the scalar product associated with W:

$$\Phi_n' W \Phi_m = \delta_{nm}.$$

The functions $(\phi_n)_{n=1}^L$ can be approximated in the same way by:

$$\phi_n^{\mathbb{D}}(x_1, x_2) \ = \ \tfrac{1}{\gamma_n} \sigma_2^2 r_2((x_1, x_2), \mathbb{D}) W \Phi_n.$$

The expectation $E(x_1, x_2)$ is approximated by:

$$E^{\mathbb{D}}(x_1, x_2) = \sum_{n=1}^{L} \phi_n(\mathbb{D})(x_1, x_2) \left[\Phi_n' W \widetilde{Z}(\mathbb{D})\right].$$

The P process Z_2 is approximated by:

$$Z_2^{\mathbb{D}}(x_1, x_2) = \widetilde{Z}_2(x_1, x_2) - E^{\mathbb{D}}(x_1, x_2).$$

According to proposition 9, this approximation of the P process is equal to the process $\widetilde{Z}_2$ conditioned to be null on the subspace $\mathbb{D}$:

$$Z_2^{\mathbb{D}}(x_1, x_2) = \left[\widetilde{Z}_2(x_1, x_2) \mid \widetilde{Z}_2(\mathbb{D}) = 0\right].$$

The second way is to find a covariance kernel of $\widetilde{Z}_2$ such that the kernel of Z_2 is computable. If the covariance kernel of $\widetilde{Z}_2$ is of the form:

$$\sigma_2^2 r_2((x_1, x_2), (t_1, t_2)) = \sigma^2 r_{2,1}(x_1, t_1) r_{2,2}(x_2 - 0.7x_1, t_2 - 0.7t_1),$$

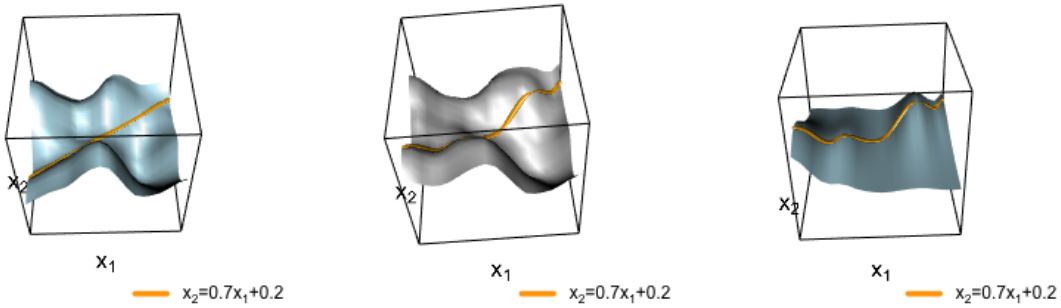
with $r_{2,1}$ and $r_{2,2}$ stationary kernels, and if Z_2 is a P process, then according to proposition 10, it is equal to:

$$Z_2(x_1, x_2) = \widetilde{Z}_2(x_1, x_2) - r_{2,2}(x_2 - 0.7x_1 - 0.2, 0)\widetilde{Z}_2(x_1, 0.7x_1 - 0.2).$$

It is a centered Gaussian process of covariance kernel $\sigma_2^2 \rho_2$ with:

$$\begin{aligned}\rho_2((x_1, x_2), (t_1, t_2)) = r_{2,1}(x_1, t_1) \quad & [r_{2,2}(x_2 - 0.7x_1, t_2 - 0.7t_1) \\ & - r_{2,2}(x_2 - 0.7x_1 - 0.2, 0) r_{2,2}(t2 - 0.7t_1 - 0.2, 0)].\end{aligned}$$

The different processes involved are shown in figure below: the P process in left panel, the latent process on middle panel, and the expectation $E(x_1, x_2) = r_{2,2}(x_2 - 0.7x_1 - 0.2, 0)\widetilde{Z}_2(x_1, 0.7x_1 + 0.2)$ on right panel.



$\boldsymbol{Z_2^P(x_1, x_2)}$ $\quad$ $\boldsymbol{\widetilde{Z}_2(x_1, x_2)}$ $\quad$ $\boldsymbol{E(x_1, x_2)}$

The example in 4D can be found in appendix 9.1.1.

3.3 Qualitative comparison of the processes

In this section, the name Classic process denotes a stationary Gaussian process. The processes Classic, Psi, Red and P (the three later are built using the Classic process as latent process) are compared in terms of path shape and influence of parameters. The processes are defined in the context of the example in 2D from subsection 3.1.2. The nullity property (see equation (3.4)) has to be verified at the line $x_2 = 0.7x_1 + 0.2$. The figures follow the same color code : the paths of the Classic process are in grey, those of the Psi process are in green, those of the Red process are in red, and those of the P process are in blue.

All the processes are centered. The different parameters at stake are σ^2 the variance parameter, $\theta = (\theta_1, \theta_2)$ the covariance parameters, and δ the additional parameter of the Psi process. The covariance kernel of the Classic process is the Matern $\frac{5}{2}$ kernel:

$$k^{Classic}((x_1, x_2), (t_1, t_2)) = \sigma^2 \prod_{i=1}^{2} (1 + \frac{\sqrt{5}\,|x_i - t_i|}{\theta_i} + \frac{5(x_i - t_i)^2}{3\theta_i^2}) \exp\left(-\frac{\sqrt{5}\,|x_i - t_i|}{\theta_i}\right).$$

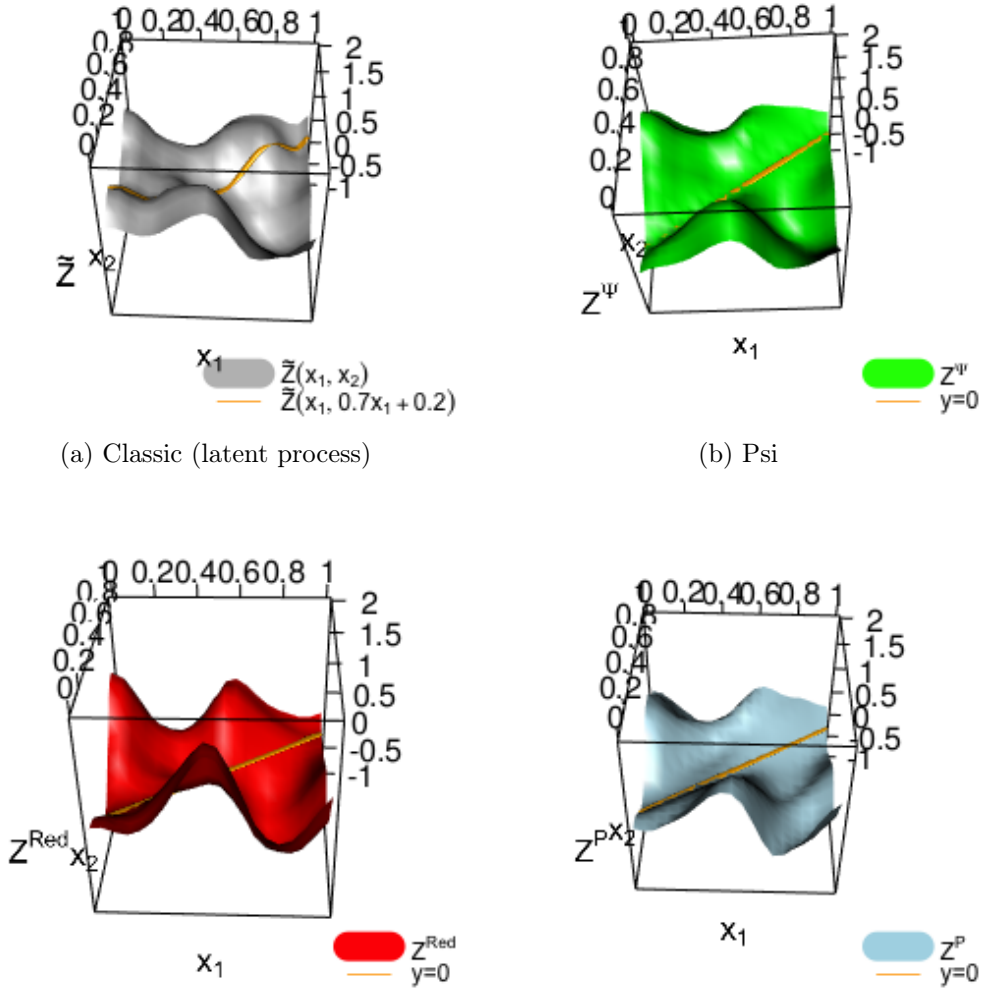
3.3.1 Shapes of the paths

Figure 3.2 shows a path of each process : Classic (see panel 3.2a), Psi (see panel 3.2b), Red (see panel 3.2c), and P (see panel 3.2d). Each path $x \mapsto Z(x, \omega)$ was taken at the same value of ω and with the same parameters : $\sigma^2 = 1$, $\theta = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$, $\delta = 0.1$. The orange line is the zone of nullity of the Psi, Red and P processes. It corresponds to the input zone $x_2 = 0.7x_1 + 0.2$. As the same ω was taken for each process, the shapes have similarities. The Red process seems to accentuate the elevations and depressions of the Classic one while the Psi and P processes keep the same order of smoothness.

A possible interpretation of that phenomenon is that the Classic process (see panel 3.2a), used as latent process for Psi, Red and P, is disturbed locally in the case of the Psi and P processes, whereas it is disturbed globally in the case of the Red process. An illustration of the processes at stake in the construction of Psi, Red and P processes is shown in figure 3.3. $\widetilde{Z}(x_1, 0.7x_1 + 0.2)$ (see panel 3.3b), which is subtracted from the latent process to build the Red process, is constant in x_2, and consequently generates a disruption of the latent process $\widetilde{Z}(x_1, x_2)$ in the whole input space $[0, 1]^2$. On the contrary, the conditional expectation $\mathbb{E}\left[\widetilde{Z}(x_1, x_2) \mid \widetilde{Z}(t_1, 0.7t_1 + 0.2),\ \forall t_1 \in [0, 1]\right]$ (see panel 3.3c), which is subtracted from the latent process to build the P process, depends on x_2, especially near the line $x_2 = 0.7x_1 + 0.2$, and tends to 0 far from the line. The modification it implies on the latent process is therefore located almost only along the line. Similarly, $\Psi(x_2 - 0.7x_1 - 0.2)$ (see panel 3.3a), which is multiplied by the latent process to build the Psi process, is equal to 0 on the line and to 1 far from the line. Although the disturbations are both local, they are of different nature: the one of the P process is smoother than the one of Psi. The P process combines the locality of the disturbation of the Psi process and the smoothness of disturbation of the Red process.

3.3.2 Influence of σ^2

Figure 3.4 shows paths of the Classic, Psi, Red, and P processes with the same parameters than in figure 3.2 except that $\sigma^2 = 3$. Consequently to the increase of σ^2, all the paths have a bigger range. The influence of σ^2 is the same for all processes. It defines the degree of dispersion of the path. The bigger σ^2 is, the more the path is dispersed.



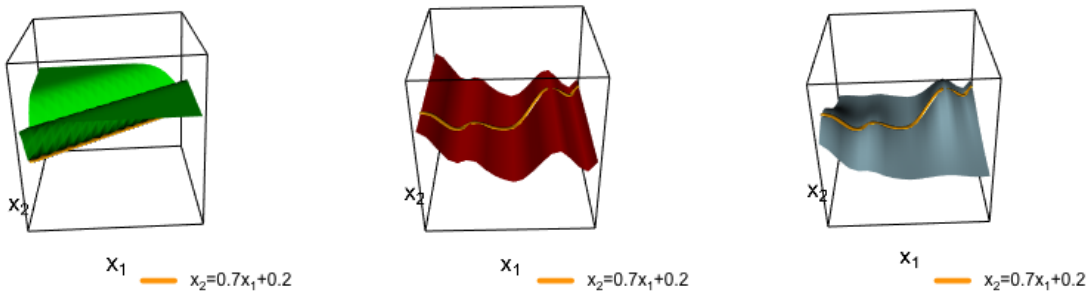
(a) Classic (latent process)

(b) Psi

(c) Red

(d) P

Figure 3.2: Comparison of Classic, Psi, Red and P paths. Visualization of a path $x \mapsto Z(x, \omega)$ of each process. The same ω is used for every process. The orange line is the line $x_2 = 0.7x_1 + 0.2$ on which the Psi, Red and P processes are null. On topleft, panel 3.2a shows a path of the Classic process. On topright, panel 3.2b shows a path of the Psi process. On bottomleft, panel 3.2c shows a path of the Red process. On bottomright, panel 3.2d shows a path of the P process.



(a) Psi function (b) Value of the latent process on the line (c) Expectation of the latent process

Figure 3.3: Processes used to transform the latent process in order to verify the nullity property. The processes on panel 3.3a, 3.3b, and 3.3c are used respectively to build the Psi, Red and P processes.

3.3.3 Influence of θ_1

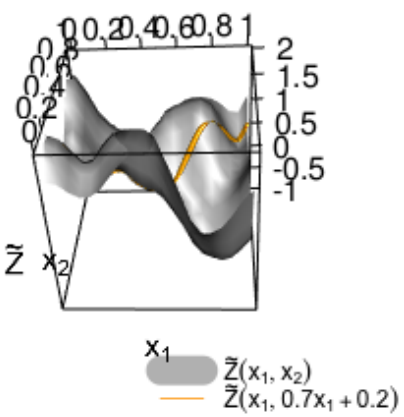
Figure 3.5 shows paths of the Classic, Psi, Red, and P processes with the same parameters than in figure 3.2 except that $\theta_1 = 0.1$. The influence of θ_1 is the same for all processes. It defines the degree of chaos of the path in the x_1 direction. The smaller θ_1 is, the more chaotic the path is in the x_1 direction.

3.3.4 Influence of θ_2

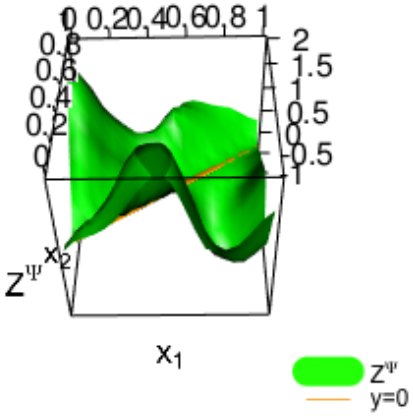
Figure 3.6 shows paths of the Classic, Psi, Red, and P processes with the same parameters than in figure 3.2 except that $\theta_2 = 0.1$. This time, the influence of θ_2 is not the same for all the processes. For the Classic and Psi processes, it defines the degree of chaos of the path in the direction x_2. For the P process, it defines the degree of chaos of the path in the direction orthogonal to the line $x_2 = 0.7x_1 + 0.2$. It is due to the fact that the variable associated to θ_2 in the covariance kernel is not x_2 but $x_2 - 0.7x_1 - 0.2$ (see subsection 3.2.3). For the Red process, θ_2 defines the degree of chaos both in x_2 and x_1 directions. It is due to the fact that both x_2 and $0.7x_1 + 0.2$ are associated with θ_2 in the covariance kernel (see subsection 3.2.1).

3.3.5 Influence of δ

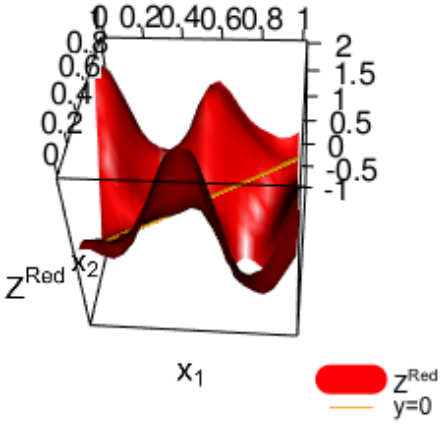
Figure 3.7 shows paths of the Psi process with $\delta = 0.05$ and $\delta = 0.3$. δ acts as a smoother of the Psi paths in the direction orthogonal to the line $x_2 = 0.7x_1 + 0.2$. Increasing δ implies a wider zone centered on the line where the path is flat near 0. In both cases, the Psi process is really disturbing the initial Classic path in the area of action delimited by δ. Indeed, for small δ, it gives a really sharp zone where the path brutally goes to 0. For big δ, it gives a really flat path in the middle, and looking like the initial Classic path only on the sides of the input space.



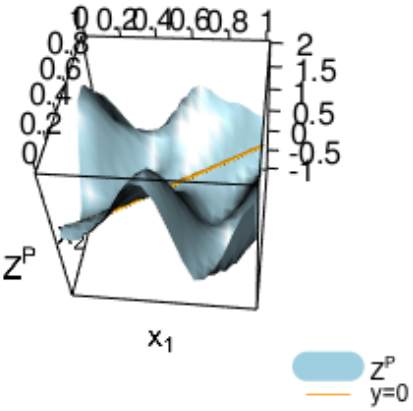
(a) Classic (latent process)



(b) Psi

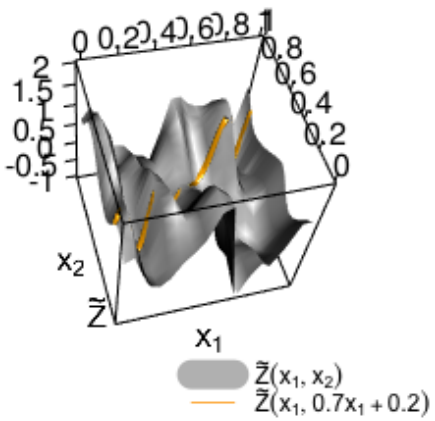


(c) Red

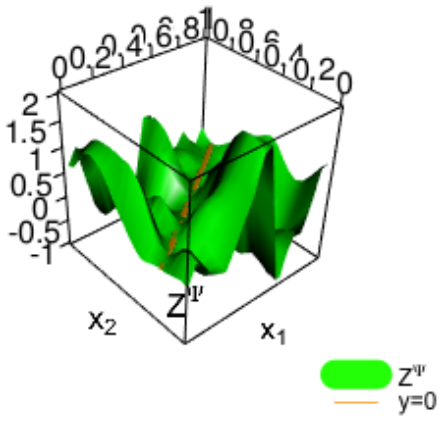


(d) P

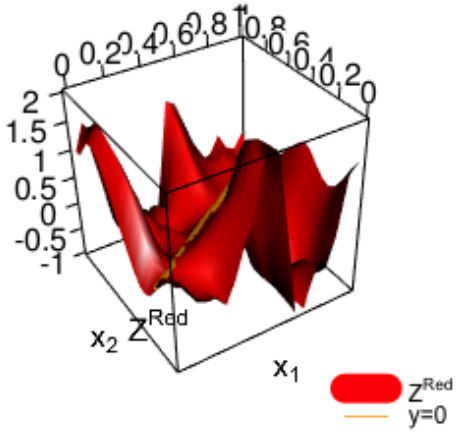
Figure 3.4: Paths of Classic, Psi, Red, and P processes for $\sigma^2 = 3$ instead of $\sigma^2 = 1$ in figure 3.2. The other parameters are unchanged : $\theta_1 = \theta_2 = \frac{\sqrt{2}}{4}$, $\delta = 0.1$.



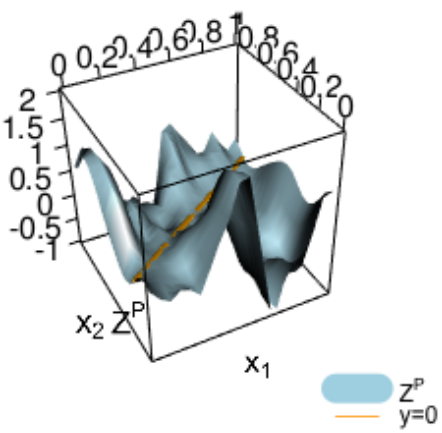
(a) Classic (latent process)



(b) Psi

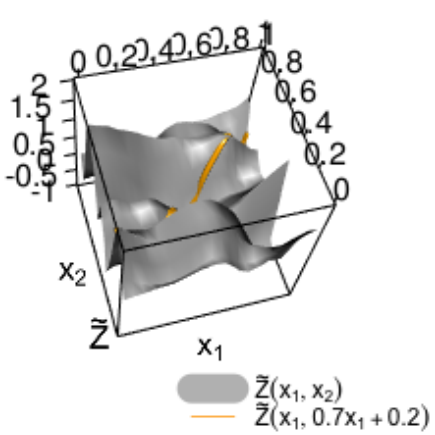


(c) Red

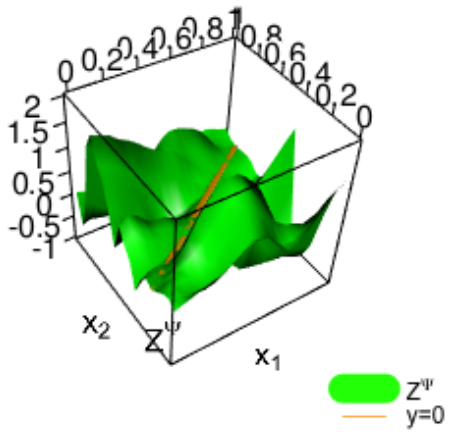


(d) P

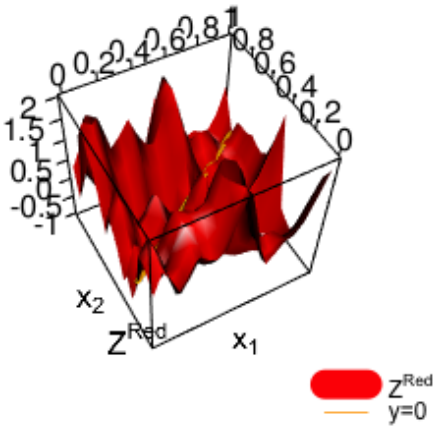
Figure 3.5: Paths of Classic, Psi, Red, and P processes for $\theta_1 = 0.1$ instead of $\theta_1 = \frac{\sqrt{2}}{4}$ in figure 3.2. The other parameters are unchanged : $\sigma^2 = 1$, $\theta_2 = \frac{\sqrt{2}}{4}$, $\delta = 0.1$.



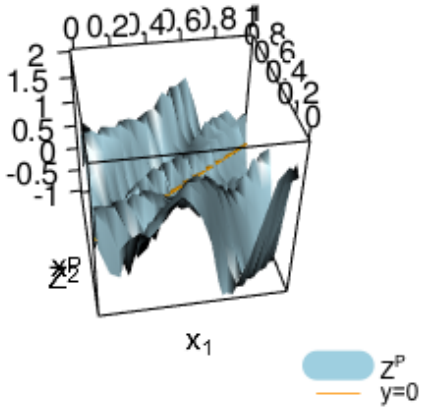
(a) Classic (latent process)



(b) Psi

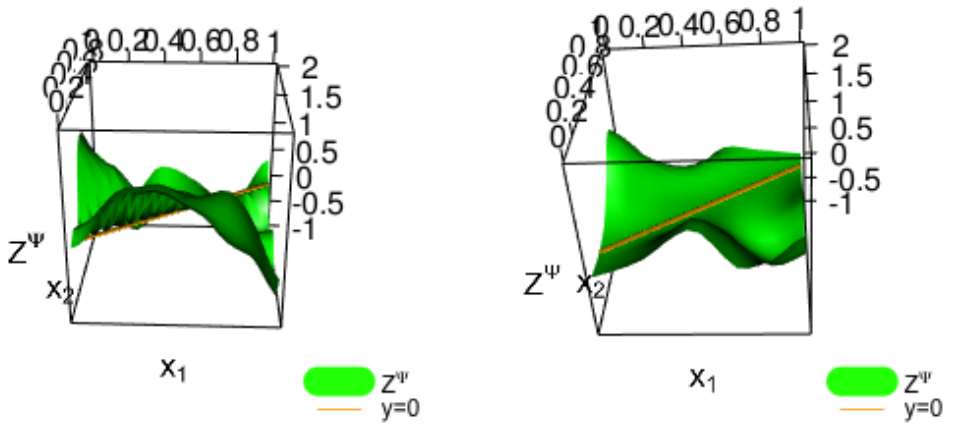


(c) Red



(d) P

Figure 3.6: Paths of Classic, Psi, Red, and P processes for $\theta_2 = 0.1$ instead of $\theta_2 = \frac{\sqrt{2}}{4}$ in figure 3.2. The other parameters are unchanged : $\sigma^2 = 1$, $\theta_1 = \frac{\sqrt{2}}{4}$, $\delta = 0.1$.



(a) $\delta = 0.05$ (b) $\delta = 0.3$

Figure 3.7: Path of the Psi process for $\delta = 0.05$ (panel 3.7a) and $\delta = 0.3$ (panel 3.7b). The other parameters are the same as in figure 3.2 : $\sigma^2 = 1$, $\theta_1 = \theta_2 = \frac{\sqrt{2}}{4}$, $\delta = 0.1$.

3.4 Estimation of the parameters

The current section aims at knowing if the candidates Psi, Red, and P can be used in pratice in the method seqGPR. A quantitative comparison is carried out between the processes Classic, Psi, Red, and P, about their performance in parameter estimation which is done by maximum likelihood. The goal is to find the process parameters from observations of these processes. These observations are generated by simulations. In the real studies, observations of the correction processes are not always available (only if the samples are nested as in subsection 4.1.1 in chapter 4).
Following subsection 2.1.2 in chapter 2, the training data is composed of simulations, denoted by $\boldsymbol{z} = (z_1, \cdots, z_{n_\mathbb{X}})'$, of the candidate process $Z \sim \mathcal{GP}(0, (\sigma_{simu})^2 \rho_{\xi_{simu}}(x,t))$ (where σ_{simu} and ξ_{simu} are the true parameters) on a set of points $\mathbb{X} = \begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(n_\mathbb{X})} \end{pmatrix}$ of size $n_\mathbb{X}$. The notation ξ_{simu} is introduced to caracterize all the covariance parameters except σ_{simu} .The parameters $\eta = (\sigma^2, \xi)$ are estimated from the training data by maximizing the likelihood $\mathcal{L}(\eta; \boldsymbol{z})$ or equivalently by minimizing the loss function $l(\xi; \mathbb{Z})$ defined by :

$$l(\xi; \mathbb{Z}) = n_\mathbb{X} \log(\sigma^2(\xi)) + \log|\rho_\xi(\mathbb{X}, \mathbb{X})|,$$

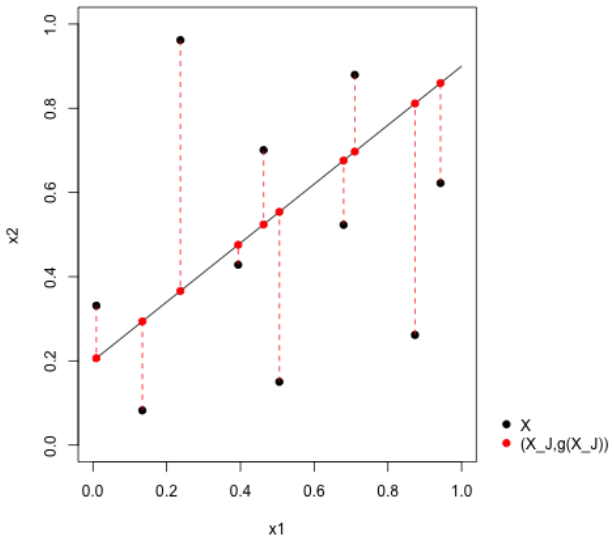
with $\sigma^2(\xi) = \frac{\boldsymbol{z}' \rho_\xi(\mathbb{X},\mathbb{X})^{-1} \boldsymbol{z}}{n_\mathbb{X}}$ the estimation of σ^2 as a function of ξ (see subsection 2.1.2 in chapter 2).

The notations $\mathbb{X}_J$, $\mathbb{X}_I$, used in the expressions of the likelihood, are explained in this paragraph. $\mathbb{X}_J$ (resp. $\mathbb{X}_I$) is the submatrix of $\mathbb{X}$ taking only the first d_J components (resp. the last d_I components) of each $x^{(i)}$. The matrix $(\mathbb{X}_J, g(\mathbb{X}_J))$ is equal to the matrix $\mathbb{X}$ where the last d_I components of each $x^{(i)}$ have been replaced by $g(x_J^{(i)})$. If $\mathbb{X} = \begin{pmatrix} x_J^{(1)} & x_J^{(1)} \\ \vdots & \\ x_J^{(n_\mathbb{X})} & x_J^{(n_\mathbb{X})} \end{pmatrix}$,

then the matrices $\mathbb{X}_J$, $\mathbb{X}_I$ and $g(\mathbb{X}_J)$ are defined as $\mathbb{X}_J = \begin{pmatrix} x_J^{(1)} \\ \vdots \\ x_J^{(n_\mathbb{X})} \end{pmatrix}$, $\mathbb{X}_I = \begin{pmatrix} x_I^{(1)} \\ \vdots \\ x_I^{(n_\mathbb{X})} \end{pmatrix}$ and $g(\mathbb{X}_J) = \begin{pmatrix} g(x_J^{(1)}) \\ \vdots \\ g(x_J^{(n_\mathbb{X})}) \end{pmatrix}$. Finally, the matrix $(\mathbb{X}_J, g(\mathbb{X}_J))$ is equal to :

$$(\mathbb{X}_J, g(\mathbb{X}_J)) = \begin{pmatrix} x_J^{(1)} & g(x_J^{(1)}) \\ \vdots & \\ x_J^{(n_\mathbb{X})} & g(x_J^{(n_\mathbb{X})}) \end{pmatrix}.$$

In the example in 2D from subsection 3.1.2, a DoE $\mathbb{X}$ is of the form $\mathbb{X} = \begin{pmatrix} x_1^{(1)} & x_2^{(1)} \\ \vdots & \vdots \\ x_1^{(n_\mathbb{X})} & x_2^{(n_\mathbb{X})} \end{pmatrix}$. The matrices $\mathbb{X}_J$, $\mathbb{X}_I$, $g(\mathbb{X}_J)$, and $(\mathbb{X}_J, g(\mathbb{X}_J))$ are equal to: $\mathbb{X}_J = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_\mathbb{X})} \end{pmatrix}$, $\mathbb{X}_I = \begin{pmatrix} x_2^{(1)} \\ \vdots \\ x_2^{(n_\mathbb{X})} \end{pmatrix}$, $g(\mathbb{X}_J) = \begin{pmatrix} 0.7x_1^{(1)} + 0.2 \\ \vdots \\ 0.7x_1^{(n_\mathbb{X})} + 0.2 \end{pmatrix}$, $(\mathbb{X}_J, g(\mathbb{X}_J)) = \begin{pmatrix} x_1^{(1)} & 0.7x_1^{(1)} + 0.2 \\ \vdots & \vdots \\ x_1^{(n_\mathbb{X})} & 0.7x_1^{(n_\mathbb{X})} + 0.2 \end{pmatrix}$. An example of samples $\mathbb{X}$ and $(\mathbb{X}_J, g(\mathbb{X}_J))$ is given on figure below:



The minimization problem resulting from the maximum likelihood estimation is given for the Psi, Red and P process. The maximum likelihood estimation is then performed for each process on the example in 2D.

3.4.1 Psi Likelihood

In the case of the Psi process, the correlation parameters are $\xi = (\theta, \delta)$. As a recall, the correlation kernel of the latent process is denoted by r_θ. The correlation matrix on the DoE

is equal to

$$\rho_{\theta,\delta}(\mathbb{X},\mathbb{X}) = diag(\Psi_\delta(\mathbb{X}_I - g(\mathbb{X}_J)))r_\theta(\mathbb{X},\mathbb{X})diag(\Psi_\delta(\mathbb{X}_I - g(\mathbb{X}_J))),$$

where $diag(\Psi_\delta(\mathbb{X}_I - g(\mathbb{X}_J)))$ is the diagonal matrix whose diagonal components are the components of the vector $\Psi_\delta(\mathbb{X}_I - g(\mathbb{X}_J)) = \left(\Psi_\delta(x_I^{(i)} - g(x_J^{(i)}))\right)_{i\in[\![1,n_\mathbb{X}]\!]}$.

The estimation problem consists in finding (θ,δ) minimizing :

$$l(\theta,\delta;\boldsymbol{z}) = n_\mathbb{X}\log\sigma^2(\theta,\delta) + \log|r_\theta(\mathbb{X},\mathbb{X})| + 2\sum_{i=1}^{n_\mathbb{X}}\Psi_\delta(x_I^{(i)} - g(x_J^{(i)})),$$

with

$$\sigma^2(\theta,\delta) = \frac{\widetilde{\boldsymbol{z}}_\delta' r_\theta(\mathbb{X},\mathbb{X})^{-1}\widetilde{\boldsymbol{z}}_\delta}{n_\mathbb{X}},$$

where $\widetilde{\boldsymbol{z}}_\delta = \left(\frac{z_i}{\Psi_\delta(x_I^{(i)} - g(x_J^{(i)}))}\right)_{i\in[\![1,n_\mathbb{X}]\!]}$.

Remark If δ is known, the problem is equivalent to minimizing the loss function of the latent process $\widetilde{Z}$ based on its observations $\widetilde{\boldsymbol{z}}_\delta$ on the sample $\mathbb{X}$.

3.4.2 Red Likelihood

In the case of the Red (Reduced) process, the correlation parameters are $\xi = \theta$. The estimation problem consists in finding θ minimizing

$$l(\theta;\boldsymbol{z}) = n_\mathbb{X}\log\left(\sigma^2(\theta)\right) + \log|\rho_\theta(\mathbb{X},\mathbb{X})|,$$

with

$$\sigma^2(\theta) = \frac{\boldsymbol{z}'\rho_\theta(\mathbb{X},\mathbb{X})^{-1}\boldsymbol{z}}{n_\mathbb{X}},$$

where

$$\rho_\theta(\mathbb{X},\mathbb{X}) = r_\theta(\mathbb{X},\mathbb{X}) - r_\theta(\mathbb{X},(\mathbb{X}_I,g(\mathbb{X}_J))) - r_\theta((\mathbb{X}_I,g(\mathbb{X}_J)),\mathbb{X}) + r_\theta((\mathbb{X}_I,g(\mathbb{X}_J)),(\mathbb{X}_I,g(\mathbb{X}_J)))).$$

3.4.3 P Likelihood

In the case of the P (Preconditioned) process, the correlation parameters are equal to $\xi = \theta$. The correlation kernel of the latent process is decomposed as : $r_\theta((x_J,x_I),(t_J,t_I)) = r_{\theta_J}(x_J,t_J)r_{\theta_I}(x_I - g(x_J),t_I - g(t_J))$, where $\theta = (\theta_J,\theta_I)$. The estimation problem consists in finding θ minimizing

$$l(\theta;\mathbb{Z}) = n_\mathbb{X}\log\left(\sigma^2(\theta)\right) + \log|\rho_\theta(\mathbb{X},\mathbb{X})|,$$

with

$$\sigma^2(\theta) = \frac{\boldsymbol{z}'\rho_\theta(\mathbb{X},\mathbb{X})^{-1}\boldsymbol{z}}{n_\mathbb{X}},$$

where

$$\begin{aligned}\rho_\theta(\mathbb{X},\mathbb{X}) &= r_{\theta_J}(\mathbb{X}_J,\mathbb{X}_J)\\ &\circ \begin{bmatrix} r_{\theta_I}(\mathbb{X}_I - g(\mathbb{X}_J),\mathbb{X}_I - g(\mathbb{X}_J)) \\ -r_{\theta_I}(\mathbb{X}_I - g(\mathbb{X}_J),0)\, r_{\theta_I}(0,\mathbb{X}_I - g(\mathbb{X}_J)) \end{bmatrix}.\end{aligned}$$

The notation $\circ$ is used to denote the pointwise product.

3.4.4 Comparison of the estimations on a $2D$ example

In this subsection, the three previously defined Gaussian processes (Psi, Red and P) are compared in terms of parameter estimation, based on a simulation of themselves (with imposed parameter values) on a training sample. This comparison is done in the context of the example in 2D from subsection 3.1.2. The goal is to recover the values of the parameters used in the simulation, which are mentioned using the subscript $simu$ (θ_{simu}, σ^2_{simu}, and δ_{simu}). The Classic process is added to the comparison as a "control" individual. The same definition of the processes as in section 3.3 is taken. In the following example, the processes are defined on $[0,1]^2$, and the Psi, Red and P processes verify the nullity property (3.4) on the line $x_2 = 0.7x_1 + 0.2$. The processes are defined as follows:

- Classic: the Classic Gaussian process of parameters σ^2_{simu}, θ_{simu}, with a covariance kernel of type stationary tensor product matern $\frac{5}{2}$ whose fomula is recalled in subsection 2.1.1 in chapter 2 and in section 3.3.
- P: the P process built on a latent process, of parameters σ^2_{simu} and θ_{simu}, derived from the Classic process, whose covariance kernel is defined as:

$$k(x,t) = \sigma^2 r_{\theta_1}(x_1,t_1) r_{\theta_2}(x_2 - 0.7x_1 - 0.2, t_2 - 0.7t_1 - 0.2),$$

  where r_{θ_1} and r_{θ_2} are 1D Matern $\frac{5}{2}$ kernels.
- Psi: the Psi process built on a latent process equal to the Classic process, of parameters σ^2_{simu}, θ_{simu}, δ_{simu}.
- Red: the Red process built on a latent process equal to the Classic process, of parameters σ^2_{simu}, θ_{simu}.

The following values are taken for the parameters : $\sigma^2_{simu} = 1$, $\theta_{simu} = \left(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4}\right)$ (called thetamin) or $\left(2\sqrt{2}, 2\sqrt{2}\right)$ (called thetamax), and $\delta_{simu} = \frac{\sqrt{2}}{4}$ (called deltamin) or $\sqrt{2}$ (called deltamax). Each process is simulated on a DoE in $[0,1]^2$ of size 10 or 100 and its parameters are reestimated by maximum likelihood on the simulated data, using the minimization problems defined in subsections 3.4.1, 3.4.2, and 3.4.3. For the Classic process, the training sample is an LHS optimized for the generalized maximin criterion with $p = 50$ (see subsection 2.1.5 in chapter 2) using the R package DiceDesign. For the other processes, the design is built as described in section 5.1 in chapter 5, such that it is well spread in $[0,1]^2$, it is sufficiently far from the line $x_2 = 0.7x_1 + 0.2$, and its projection on the x_1 direction is equal to $(\frac{i}{n_{\mathbb{X}}-1})_{i \in [\![0, n_{\mathbb{X}}-1]\!]}$ (with $n_{\mathbb{X}} = 10$ or 100).

The boxplots of the parameter estimators are computed other 100 simulations for each training size. The boxplots corresponding to an estimation over the Classic, Psi, Red and P processes are respectively in grey, green, red and blue. A distinction is made for the Psi process between an estimation when $\delta = \frac{\sqrt{2}}{4}$ (called Psi_deltamin in lightgreen) and $\delta = \sqrt{2}$ (called Psi_deltamax in green).

θ estimation Figure 3.8 shows boxplots of estimated values of θ_1 and θ_2 for the two sizes of the training sample. All estimations are centered on the true values θ_{simu} and become more accurate when the training size increases. This result seems to indicate that the maximum likelihood estimators converge towards the true values when the sample size goes to $+\infty$. For a training size of 100, all processes have similar performances. For a training size of 10, the

Classic process is more robust than the others. This can be due to the fact that this process is stationary, but also that the DoE's are better spread in the input space (because subject to less constraints in their building). Red and P are comparable and Psi (Psi_deltamin or Psi_deltamax) is a bit worse than the others. This result may be explained by the fact that the Psi process has one more parameter to estimate and that can disturb the estimation.

σ^2 estimation Figure 3.9 shows boxplots of estimated values of σ^2 for the two sizes of the design. All estimations seem to converge except in the case $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4}), \delta_{simu} = \sqrt{2}$. Furthermore, the estimation of σ^2 is really bad for a training size of 10 in the case of the Psi process, compared to the other processes.

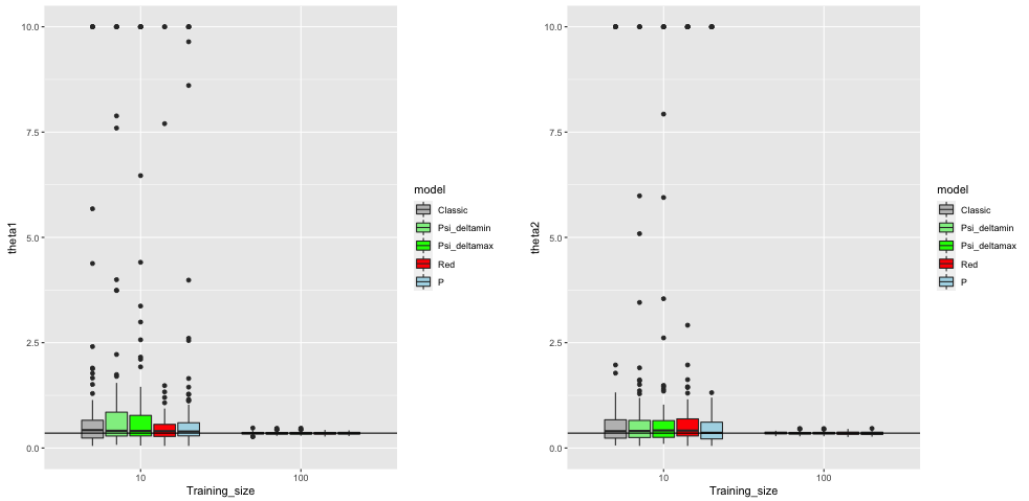
δ estimation for Psi Figure 3.10 shows boxplots of estimated values of δ for the different training sizes and values of δ_{simu} and θ_{simu}. In all cases except $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4}), \delta_{simu} = \sqrt{2}$, the estimation of δ seems to converge towards the true value δ_{simu}. For this particular case when it does not converge, the estimations are very scattered from 0 to 10. The problem is probably that δ and θ play similar roles of caracteristic lengths and that may make their separate estimation impossible.

3.5 Conclusion

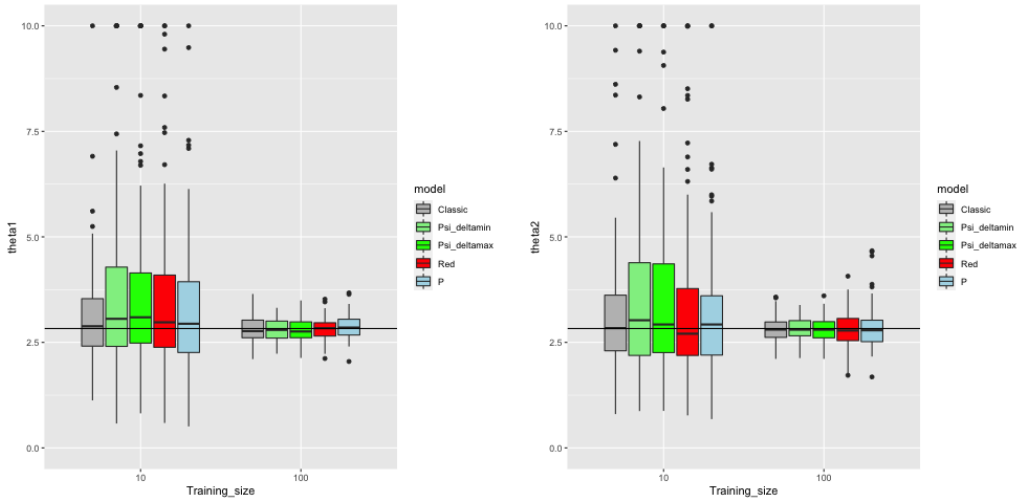
This chapter defines the underlying Gaussian processes and their relations in the seqGPR metamodel: a Gaussian process Y_1 modeling the training data at step 1, and correction processes modeling the additional pieces of information brought about by the successive steps. The different processes are assumed independent to simplify calculations. This hypothesis is necessary and not directly implied by their definition. The correction processes must be Gaussian processes null on an infinite continuous set of points.

Three candidates are defined for the correction processes: Red, Psi and P. They result from transformations of a latent process whose kernel is numerically computable. The Red process is the most simple candidate. It is equal to the latent process reduced by its value on the subspace of nullity. This transformation disturbs the value of the latent process in the entire input space. Therefore, a second candidate is suggested, called Psi, which consists in multiplying the latent process by a function null on the subspace of nullity. This transformation of the latent process is local (near the nullity subspace), but not as smooth as the previous one, as the Psi process is equal to the latent process far from the nullity zone, but is forced to go to zero on this zone. Finally, a last candidate is proposed, called P, which is nearer to the Red process. The difference is that the latent process is not reduced directly by its value on the nullity subspace, but by its expectation. The expectation being dependent on all the inputs, the transformation is smoother than for Psi but stays local, and far from the nullity subspace, the P process is equal to the latent process. The problem of this process is that its kernel is not always numerically computable. Two methods are tried to obtain that computability. The first one, consisting in discretizing its spectral decomposition is not retained, as it does not preserve the nullity in the whole nullity subspace. The second method is prefered, which directly gives an analytical formula for a particular choice of the latent process kernel.

The parameter influences on the path shapes of the candidates and the Classic process (which is used as latent process to build them) are compared in the 2D example. The shapes are

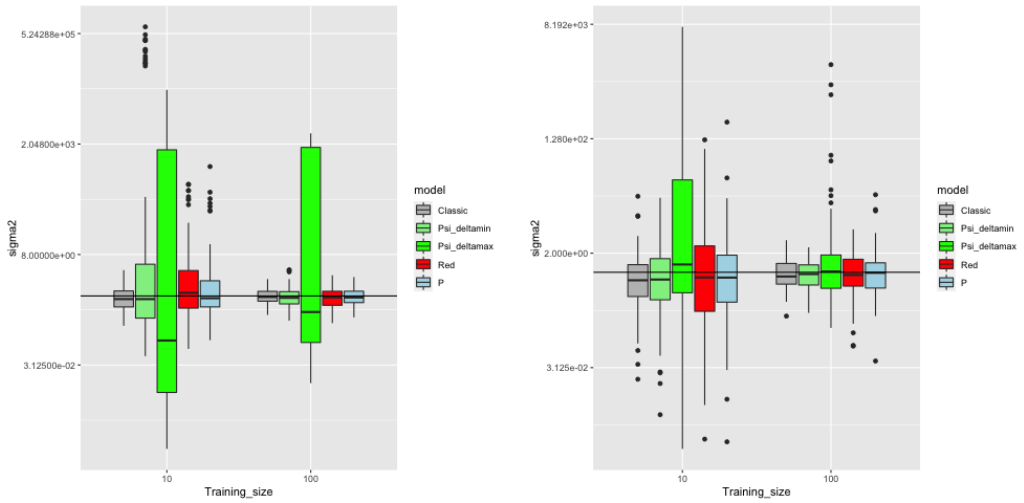


(a) Estimation of θ_1 with $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$ (b) Estimation of θ_2 with $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$



(c) Estimation of θ_1 with $\theta_{simu} = (2\sqrt{2}, 2\sqrt{2})$(d) Estimation of θ_2 with $\theta_{simu} = (2\sqrt{2}, 2\sqrt{2})$

Figure 3.8: Estimation, on 100 training samples, of the covariance parameters of the different processes (Classic, Psi, Red, P) : θ_1 (panels 3.8a and 3.8c) and θ_2 (panels 3.8b and 3.8d). Different values of θ are used in the DoE : $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$ (in panels 3.8a and 3.8b), or $\theta_{simu} = (2\sqrt{2}, 2\sqrt{2})$ (in panels 3.8c and 3.8d), along with different sizes of training sample (10 or 100). The variance parameter σ^2_{simu} is set to 1. In the case of the Psi process, the parameter δ_{simu} is set to $\frac{\sqrt{2}}{4}$ (legend Psi_deltamin) or $\sqrt{2}$ (legend Psi_deltamax). In each boxplot, the black horizontal line is the value of θ_{simu}. The estimated value of θ results from the joint estimation of $(\theta, \sigma^2, \delta)$ by maximization of the likelihood.

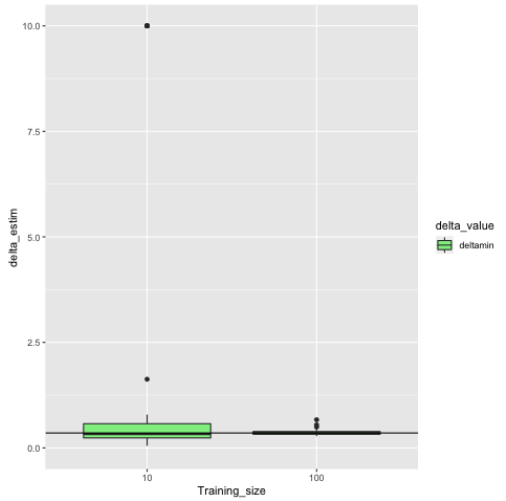


(a) $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$ (b) $\theta_{simu} = (2\sqrt{2}, 2\sqrt{2})$

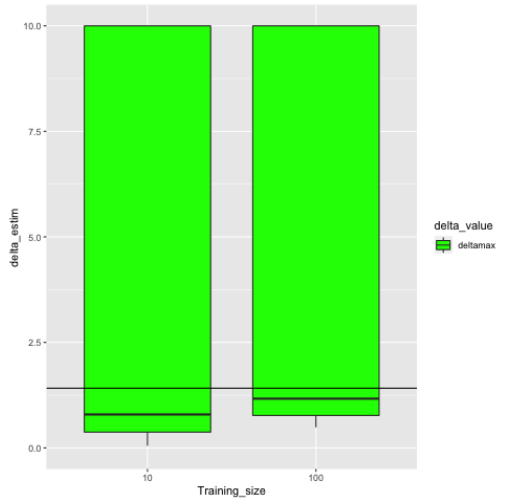
Figure 3.9: Estimation of the variance parameter σ^2. The true value σ^2_{simu}, represented by the black horizontal line, is set to 1. θ_{simu} is set to $(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$ (panel 3.9a) or $(2\sqrt{2}, 2\sqrt{2})$ (panel 3.9b). δ_{simu} is set to $\frac{\sqrt{2}}{4}$ or $\sqrt{2}$. For each training size (10 or 100), from left to right are represented the boxplots of the following processes : Classic, Psi (Psi_deltamin when $\delta_{simu} = \frac{\sqrt{2}}{4}$ and Psi_deltamax when $\delta_{simu} = \sqrt{2}$), Red and P. The y-axis is plotted on a logarithmic scale.

similar, the one of the Red process is more different as the transformation concerns the entire input space. The influence of σ^2 and θ_1 are the same for all processes: σ^2 determines the dispersion of the path, and θ_1 monitors the smoothness in the x_1 direction. θ_2 has a different meaning depending on the process. For Classic and Psi, it influences the smoothness of the path in the x_2 direction, for P it influences the smoothness in the direction orthogonal to the nullity subspace, for Red it influences the smoothness in x_1 and x_2 directions. δ determines the speed at which the Psi process goes from 0 to the value of the latent process.

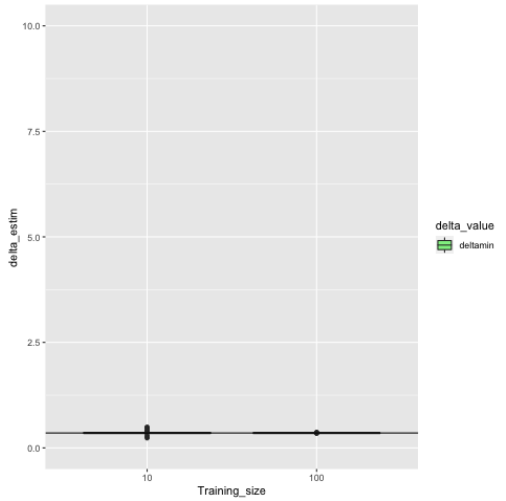
It seems that the maximum likelihood estimation of the parameters of the Psi process is not robust (especially for the δ parameter), even in a simple case like the one of the 2D example (with only two inputs and a big training sample of size 100). Furthermore, a big dimension implies a lot more parameters to estimate than the other processes. The estimation of the Red and P parameters is of good quality: it converges with the size of the DoE and it is of the same order of magnitude as the estimation of the Classic process parameters. The latter is a little bit more robust because Red and P processes are more complicated as they are not second-order stationary, but they seem viable. From these observations, only the processes Red and P, whose paths are smoother and whose parameter estimation is robust, are kept to build the metamodel seqGPR.



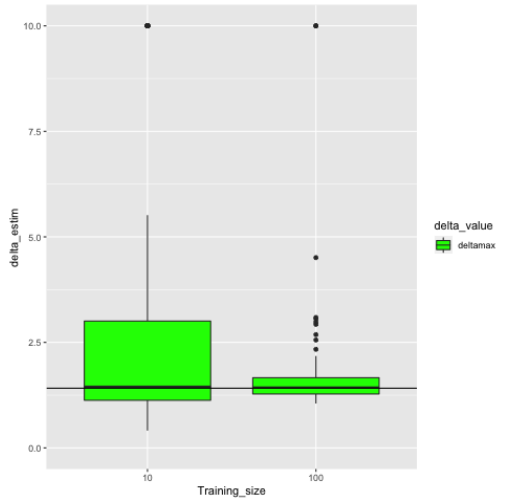
(a) $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$,$\delta_{simu} = \frac{\sqrt{2}}{4}$



(b) $\theta_{simu} = (\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$,$\delta_{simu} = \sqrt{2}$



(c) $\theta_{simu} = (2\sqrt{2}, 2\sqrt{2})$,$\delta_{simu} = \frac{\sqrt{2}}{4}$



(d) $\theta_{simu} = (2\sqrt{2}, 2\sqrt{2})$,$\delta_{simu} = \sqrt{2}$

Figure 3.10: Estimation of δ for the Psi process. The true value δ_{simu}, represented by the black horizontal line, is set to $\frac{\sqrt{2}}{4}$ (panels 3.10a and 3.10c) or $\sqrt{2}$ (panels 3.10b and 3.10d). θ_{simu} is set to $(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{4})$ (panels 3.10a and 3.10b) or $(2\sqrt{2}, 2\sqrt{2})$ (panels 3.10c and 3.10d). σ^2_{simu} is set to 1. The training sizes are equal to 10 or 100.

Chapter 4

seqGPR methodology

Chapter 3 proposes the methodology seqGPR taking into account results from numerical simulations $\boldsymbol{y_1}$, ..., $\boldsymbol{y_N}$, obtained from DoE's $\mathbb{X}_1$, ..., $\mathbb{X}_N$ in spaces of increasing dimension $[0,1]^{d_1}$, ..., $[0,1]^{d_1+\cdots+d_N}$ (see subsection 3.1.1 in chapter 3). More precisely, $\boldsymbol{y_1}$, ..., $\boldsymbol{y_N}$ are assumed to be the realizations of the processes Y_1, ..., Y_N at the designs $\mathbb{X}_1$, ..., $\mathbb{X}_N$, where (for $x_{I_1\cup\cdots\cup I_n}$ in $[0,1]^{d_1+\cdots+d_n}$):

$$\begin{cases} Y_1(x_{I_1}) & = & m + Z_1(x_{I_1}), \\ \\ Y_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) & = & Y_{n-1}(x_{I_1\cup\cdots\cup I_{n-1}}) + Z_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}), \quad \forall n \in [\![2, N]\!], \end{cases} \tag{4.1}$$

where:

- The processes $(Z_n)_{n=1}^N$ are independent Gaussian processes of law:

$$Z_n \sim \mathcal{GP}(0, \sigma_n^2 \rho_n(x_{I_1\cup\cdots\cup I_n}, t_{I_1\cup\cdots\cup I_n})), \quad \forall n \in [\![1, N]\!]. \tag{4.2}$$

- The processes $(Z_n)_{n=2}^N$ verify the following property:

$$Z_n(x_{I_1\cup\cdots\cup I_{n-1}}, \grave{x}_{I_n}) = 0, \quad \forall n \in [\![2, N]\!], \forall x_{I_1\cup\cdots\cup I_{n-1}} \in [0,1]^{d_1+\cdots+d_{n-1}}. \tag{4.3}$$

The issue of defining the correction processes $(Z_n)_{n=2}^N$ as Gaussian processes verifying the nullity property (4.3) is solved in chapter 3 where two candidates are retained: the Red and P processes. This new chapter deals with the estimation of the parameters of the processes involved in the model (4.1), and the prediction of the model.

The law of Y_1 (see equations (4.1) and (4.2)) depends on the parameters: m (scalar mean parameter), σ_1^2 (scalar variance parameter), and θ_1 (vector of covariance parameters). For all $n \in [\![2, N]\!]$, Z_n (see equation (4.2)) is either a Red or P process built on the latent process $\widetilde{Z}_n \sim \mathcal{GP}(0, \sigma_n^2 r_n)$ (see chapter 3) of parameters: σ_n^2 (scalar variance parameter), θ_n (vector of covariance parameters). In order to emphasize the dependence of ρ_n (the covariance kernel of Z_n, $n \in [\![1, N]\!]$) on the parameter θ_n, this kernel is denoted by ρ_{θ_n}. The purpose of this chapter is to estimate the parameters $\eta = (\underbrace{m, \sigma_1^2, \theta_1}_{\eta_1}, \underbrace{\sigma_2^2, \theta_2}_{\eta_2}, \cdots, \underbrace{\sigma_n^2, \theta_n}_{\eta_n}, \cdots, \underbrace{\sigma_N^2, \theta_N}_{\eta_N})$ based on the observed data. The maximum likelihood estimator is used. Using the notations $\boldsymbol{y}' = (\boldsymbol{y_1}', \cdots, \boldsymbol{y_N}')$ and $\boldsymbol{Y}' = (Y_1(\mathbb{X}_1)', \cdots, Y_N(\mathbb{X}_N)')$, the following loss function [1] has to be minimized:

$$l(\eta; \boldsymbol{y}) = \log |Cov_\eta(\boldsymbol{Y}, \boldsymbol{Y})| + (\boldsymbol{y} - \mathbb{E}_\eta[\boldsymbol{Y}])' Cov_\eta(\boldsymbol{Y}, \boldsymbol{Y})^{-1} (\boldsymbol{y} - \mathbb{E}_\eta[\boldsymbol{Y}]).$$

[1] equal to twice the negative log-likelihood up to a constant

This optimization problem is complex as η can reach big dimensions.

To optimize this loss function, similarly to what occurs in a mutifidelity context, two cases may be distinguished. When the designs are nested [Le Gratiet and Garnier, 2014], observations of Y_1, Y_2 ..., Y_N imply observations of Y_1, Z_2, ..., Z_N. The likelihood can be decoupled into independent parts to be optimized in small dimension. For each Z_n, the estimation of its parameters is reduced to the case treated in section 3.4 in chapter 3. The case with nested designs is tackled in subsection 4.1.1. When the designs are non-nested [Zertuche, 2015], the loss function cannot be decoupled and an EM (Expectation-Maximization, see subsection 2.1.2 in chapter 2) algorithm is used instead. In subsection 4.1.2, the EM algorithm is adapted to the seqGPR metamodel, in order to optimize the joint loss function while taking advantage of the underlying additive model involving independent processes. In section 4.2, the seqGPR metamodel is tested on two analytic and one industrial test cases.

4.1 Estimation and prediction

This section shows estimation and prediction methods for the seqGPR metamodel in case of nested or non-nested designs.

4.1.1 Nested designs

Definition 7 (Nested designs) *In what follows, the notation $\mathbb{X}_n \sqsubset \mathbb{X}_{n-1}$ means that the submatrix of $\mathbb{X}_n$ composed of the columns corresponding to $x_{I_1 \cup \cdots \cup I_{n-1}}$ is included in $\mathbb{X}_{n-1}$. In this case, the designs $\mathbb{X}_n$ and $\mathbb{X}_{n-1}$ are said nested.*

Proposition 11 *If all the designs are nested: $\mathbb{X}_N \sqsubset \cdots \sqsubset \mathbb{X}_1$ (see definition 7), then the loss function can be decoupled:*

$$l(\eta; \boldsymbol{y}) = l_1(\eta_1; \boldsymbol{y_1}) + \sum_{n=2}^{N} l_n(\eta_n; \boldsymbol{z_n}),$$

with $l_1(\eta_1; \boldsymbol{y_1})$ (respectively $l_n(\eta_n; \boldsymbol{z_n})$, $n \in [\![2, N]\!]$) the loss function associated with the training data $Y_1(\mathbb{X}_1) = \boldsymbol{y_1}$ (respectively $Z_n(\mathbb{X}_n) = \boldsymbol{z_n}$).

Proof *$(\boldsymbol{z_n})_{2 \leqslant n \leqslant N}$ is observed because the designs are nested. The loss function can be decoupled because $(Y_1, Z_2, \cdots, Z_N)$ are independent.* ∎

If all the designs are nested, the parameters η_n can be estimated separately by optimizing the loss functions l_n. The formulae of the loss functions $(l_n)_{n=2}^N$ are given in subsections 3.4.2 or 3.4.3 depending whether Z_n is a Red or P process. The formula of l_1 is given in subsection 2.1.2 in chapter 2.

The same conclusion can be done for the prediction mean and variance:

Proposition 12 *If all the designs are nested: $\mathbb{X}_N \sqsubset \cdots \sqsubset \mathbb{X}_1$ (see definition 7), the prediction formulae can be decoupled:*

$$\begin{cases} \hat{y}(x) &= \mathbb{E}\left[Y_1(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}\right] + \displaystyle\sum_{n=2}^{N} \mathbb{E}\left[Z_n(x) \mid Z_n(\mathbb{X}_n) = \boldsymbol{z_n}\right], \\ \hat{v}(x) &= Var\left[Y_1(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}\right] + \displaystyle\sum_{n=2}^{N} Var\left[Z_n(x) \mid Z_n(\mathbb{X}_n) = \boldsymbol{z_n}\right]. \end{cases}$$

Proof *Same arguments as for proposition 11.* ∎

In fact, the metamodel is the sum of N independent metamodels.

Remark (properties of nested designs) To optimize the likelihoods, the matrices $cov(Z_n(\mathbb{X}_n), Z_n(\mathbb{X}_n)) = \sigma_n^2 \rho_{\theta_n}(\mathbb{X}_n, \mathbb{X}_n)$ (n in $[\![1, N]\!]$) must be inverted. First, for all n in $[\![1, N]\!]$, the design $\mathbb{X}_n$ is assumed not having any redundant points such that the associated covariance matrix has no identical rows. Furthermore, the points must be sufficiently far from each other to ensure the good conditioning of the matrix. This condition is sufficient for the first matrix as long as ρ_{θ_1} is a positive definite kernel. As it is not the case for the $(\rho_{\theta_n})_{n=2}^N$, to ensure that the matrices are not singular, each $\mathbb{X}_n$ (n in $[\![2, N]\!]$) is also supposed to not contain any point belonging to the nullity zone $\{(x_{I_1 \cup \cdots \cup I_{n-1}}, \dot{x}_{I_n}),\ x_{I_1 \cup \cdots \cup I_{n-1}} \in [0,1]^{d_1 + \cdots + d_{n-1}}\}$ so there are no rows full of zeros in the matrices. Section 5.1 in chapter 5 describes the building method retained for the samples in this context.

Drawbacks This nesting constraint on the designs implies that the size of $\mathbb{X}_n$ cannot increase with n. This is problematic as conversely the dimension of the input space increases with n, so more and more information should be brought as n increases. Furthermore, this limits the exploration of the first dimensions as the same values are used in all the designs. That is why an alternative is proposed in the next subsection with designs that are subject to less radical constraints.

4.1.2 Non-nested designs

In this subsection, the designs are said non-nested if for all n in $[\![1, N]\!]$, the $x_{I_1 \cup \cdots \cup I_n}$ part of $\mathbb{X}_n$, $\cdots$, $\mathbb{X}_N$ have no points in common. If the designs are non-nested, the loss function is not decoupled and therefore is difficult to optimize. An EM (Expectation-Maximization, see subsection 2.1.2 in chapter 2) algorithm is then used to reduce the dimension of the optimization problem. To define this algorithm, the notion of complete data is introduced.

Definition 8 (Complete data) *Let $\widetilde{\mathbb{X}}_n$ ($\forall 1 \leqslant n \leqslant N$) denote the union of all designs that concern Z_n. $\widetilde{\mathbb{X}}_n$ is a matrix of $d_1 + \cdots + d_n$ columns, composed of the concatenation of the parts of $\mathbb{X}_n, \cdots, \mathbb{X}_N$ corresponding to the input variables $x_{I_1 \cup \cdots \cup I_n}$. The complete data is the random vector: $T = (Y_1(\widetilde{\mathbb{X}}_1), Z_2(\widetilde{\mathbb{X}}_2), \cdots, Z_N(\widetilde{\mathbb{X}}_N))$.*

In the case of non-nested designs $\widetilde{\mathbb{X}}_1, \cdots, \widetilde{\mathbb{X}}_N$ have no redundant points and are nested in each other, so they play the role of $\mathbb{X}_1, \cdots, \mathbb{X}_N$ in the case of nested designs. According to proposition 11, if the values of Y_1 on $\widetilde{\mathbb{X}}_1$, denoted by z_1, and the values of the Z_n on the $\widetilde{\mathbb{X}}_n$ ($n \in [\![2, N]\!]$), denoted by z_n, were observed, the loss function associated to all those DoE's (equal to twice the negative loglikelihood of the complete data up to a constant), denoted by l_c, could be decomposed as

$$l_c(\eta; z_1, \cdots, z_n) = \sum_{n=1}^{N} l_c^n(\eta_n; z_n),$$

where $l_c^1(\eta_1; \boldsymbol{z}_1)$ is the loss function associated to the training data $Y_1(\widetilde{\mathbb{X}}_1) = z_1$, and $l_c^n(\eta_n; \boldsymbol{z}_n)$ ($n \in [\![2, N]\!]$) is the loss function associated to the training data $Z_n(\widetilde{\mathbb{X}}_n) = z_n$.

As some of the complete data is not observed (missing data), an EM algorithm seems adequate to optimize the loss function.

Definition 9 (EM algorithm) *The EM algorithm is defined by*

- ***Expectation*** *Instead of the observed data loss function, the expectation of the complete data loss function conditioned by the observed data is considered. This quantity can be decomposed in N terms:*

$$\begin{aligned} \mathcal{Q}(\eta, \eta^*) &= \mathop{\mathbb{E}}_{N}{}_{\eta^*} \left[l_c(\eta; T) \mid \boldsymbol{Y} = \boldsymbol{y} \right], \\ &= \sum_{n=1}^{N} \mathcal{Q}_n(\eta_n; \eta^*), \end{aligned}$$

  with $\forall n \in [\![1, N]\!]$

$$\mathcal{Q}_n(\eta_n, \eta^*) = \mathbb{E}_{\eta^*} \left[l_c^n(\eta_n; Z_n(\widetilde{\mathbb{X}}_n)) \mid \boldsymbol{Y} = \boldsymbol{y} \right].$$

- ***Maximization*** *The EM algorithm consists in building the sequence $(\eta^{(i)})_{i \geqslant 0} = (\eta_1^{(i)}, \cdots, \eta_N^{(i)})_{i \geqslant 0}$ such that $\eta^{(0)}$ is user defined and $\forall i \geqslant 0$, $\eta^{(i+1)}$ is solution of the optimization problem:*

$$\min_{\eta} \mathcal{Q}(\eta, \eta^{(i)}).$$

  For all $n \in [\![1, N]\!]$, $\eta_n^{(i+1)}$ is solution of the following optimization problem:

$$\min_{\eta_n} \mathcal{Q}_n(\eta_n, \eta^{(i)}). \tag{4.4}$$

See appendix 9.4.1 for the explicit formulae of the Q_n. In the usual EM algorithm (see subsection 2.1.2 in chapter 2), the maximization step consists in a maximization problem because the $\mathcal{Q}$ criterion is built on the loglikelihood. However, in this case, the $\mathcal{Q}$ criterion is built on twice the negative loglikelihood, that is why the optimization problem becomes a minimization one. Finally, η is estimated by a term of the sequence of sufficiently high rank, the choice of the rank being done by imposing a maximum number of iterations and a threshold on the variation of the loss function between iterations.

Remark (properties of non-nested designs) To optimize $\mathcal{Q}_n$, the matrices $cov\left(Z_n(\widetilde{\mathbb{X}}_n), Z_n(\widetilde{\mathbb{X}}_n)\right) = \sigma_n^2 \rho_{\theta_n}(\widetilde{\mathbb{X}}_n, \widetilde{\mathbb{X}}_n)$ $(n \in [\![1, N]\!])$ need to be inverted. First, for all $n \in [\![1, N]\!]$, $\widetilde{\mathbb{X}}_n$ is assumed not having any redundant points such that the matrices have no identical rows. This condition is sufficient for the first matrix to be invertible as long as ρ_{θ_1} is a positive-definite kernel. As it is not the case of the ρ_{θ_n} $(n \geqslant 2)$, to ensure that the matrices are not singular, each $\widetilde{\mathbb{X}}_n$ $(n \in [\![2, N]\!])$ is also supposed not to contain any point in the nullity subspace $\{(x_{I_1 \cup \cdots \cup I_{n-1}}, \grave{x}_{I_n}),\ x_{I_1 \cup \cdots \cup I_{n-1}} \in [0, 1]^{d_1 + \cdots + d_{n-1}}\}$ so there are no rows full of zeros in the matrices. Section 5.2 in chapter 5 describes the building method retained for the samples satisfying these constraints.

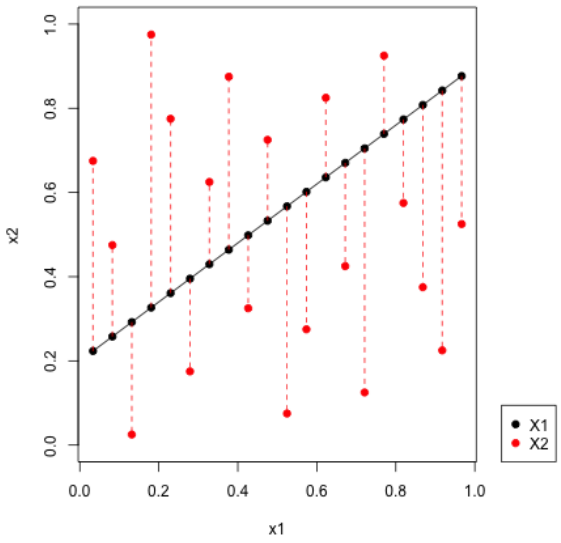
4.1.3 Example in 2D

This section illustrates the building methods of the seqGPR metamodel in the case of the example in 2D defined in subsection 3.1.2 in chapter 3.

Nested designs The two samples $\mathbb{X}_1$ and $\mathbb{X}_2$ are assumed nested (denoted by $\mathbb{X}_2 \sqsubset \mathbb{X}_1$, see definition 7 in subsection 4.1.1), i.e. they are of the form:

$$\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}, \ \mathbb{X}_2 = \begin{pmatrix} x_1^{(i_1)} & x_2^{(i_1)} \\ \vdots & \\ x_1^{(i_{n_2})} & x_2^{(i_{n_2})} \end{pmatrix}, \ \{i_1, \cdots, i_{n_2}\} \subset \{1, \cdots, n_1\}.$$

The values of x_1 contained in $\mathbb{X}_2$ are also contained in $\mathbb{X}_1$. An example of such samples is shown in figure below:



If $\mathbb{X}_1$ and $\mathbb{X}_2$ are nested, then the values of Y_1 and Z_2 on $\mathbb{X}_2$ can be deduced. Indeed, let denote by $\boldsymbol{y_1} = \begin{pmatrix} y_1^{(1)} \\ \vdots \\ y_1^{(n_1)} \end{pmatrix}$ and $\boldsymbol{y_2} = \begin{pmatrix} y_2^{(1)} \\ \vdots \\ y_2^{(n_2)} \end{pmatrix}$ the components of $\boldsymbol{y_1}$ and $\boldsymbol{y_2}$, and let $(x_1^{(i_k)}, x_2^{(i_k)})$ denote a point from $\mathbb{X}_2$, then:

$$\underbrace{Y_2(x_1^{(i_k)}, x_2^{(i_k)})}_{=y_2^{(k)}} = \underbrace{Y_1(x_1^{(i_k)})}_{=y_1^{(i_k)}} + \underbrace{Z_2(x_1^{(i_k)}, x_2^{(i_k)})}_{=z_2^{(k)}}.$$

The value of $Y_2(x_1^{(i_k)}, x_2^{(i_k)})$ is deduced by taking the kth term of the vectorial equality $Y_2(\mathbb{X}_2) = \boldsymbol{y_2}$. The value of $Y_1(x_1^{(i_k)})$ is deduced by taking the $i_k th$ term from the vectorial equality $Y_1(\mathbb{X}_1) = \boldsymbol{y_1}$, because $x_1^{(i_k)}$ is the x_1 component of the kth point of $\mathbb{X}_2$ but, as $\mathbb{X}_2 \sqsubset \mathbb{X}_1$ (see definition 7 in subsection 4.1.1), it is also the x_1 component of the $i_k th$ point of $\mathbb{X}_1$. The value of $Z_2(x_1^{(i_k)}, x_2^{(i_k)})$ is then deduced by subtraction: $z_2^{(k)} = y_2^{(k)} - y_1^{(i_k)}$. Thus, a new training data is known:

$$\begin{cases} Y_1(\mathbb{X}_1) &= \boldsymbol{y_1} \\ Z_2(\mathbb{X}_2) &= \boldsymbol{z_2}, \end{cases}$$

with $\boldsymbol{z_2}$ the vector whose kth component is equal to $z_2^{(k)}$. The likelihood to maximize becomes (with $h_V(\boldsymbol{v})$ denoting the density of random vector V taken at $\boldsymbol{v}$):

$$\begin{aligned} h_Y(\boldsymbol{y}) &= h_{Y_1(\mathbb{X}_1), Y_2(\mathbb{X}_2)}(\boldsymbol{y_1}, \boldsymbol{y_2}) \\ &= h_{Y_1(\mathbb{X}_1), Z_2(\mathbb{X}_2)}(\boldsymbol{y_1}, \boldsymbol{z_2}) \\ &= h_{Y_1(\mathbb{X}_1)}(\boldsymbol{y_1}) h_{Z_2(\mathbb{X}_2)}(\boldsymbol{z_2}). \end{aligned}$$

The last equality is explained by the independence of Y_1 and Z_2. As a consequence, the loss function $l(\eta; \boldsymbol{y})$, equal to $-2\log(h_Y(\boldsymbol{y}))$ up to a constant, is decomposed as the sum of the two loss functions associated with $Y_1(\mathbb{X}_1)$ and $Z_2(\mathbb{X}_2)$:

$$l(\eta; \boldsymbol{y}) = l_1(\eta_1; \boldsymbol{y_1}) + l_2(\eta_2; \boldsymbol{z_2}).$$

These two loss functions can be optimized separately, which leads to two simple optimization problems instead of one complex optimization problem. Thus, the parameter estimation is made easier. The same decoupling property is observed for the prediction:

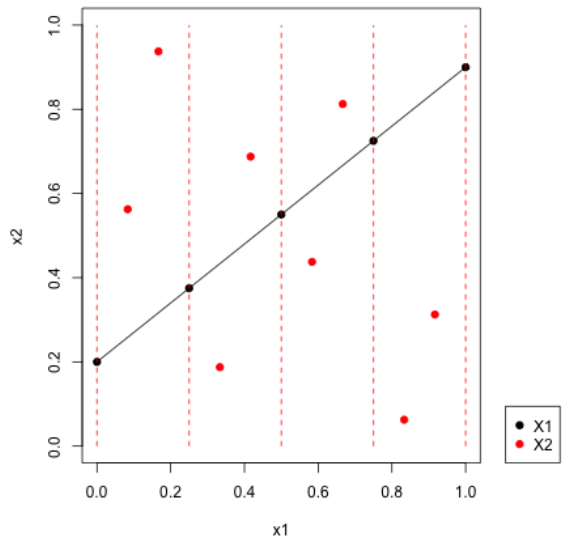
$$\left\{ \begin{array}{rcl} \hat{y}(x_1, x_2) & = & \mathbb{E}\left[Y_1(x_1) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}\right] + \mathbb{E}\left[Z_2(x_1, x_2) \mid Z_2(\mathbb{X}_2) = \boldsymbol{z_2}\right], \\ \hat{v}(x_1, x_2) & = & Var\left(Y_1(x_1) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}\right) + Var\left(Z_2(x_1, x_2) \mid Z_2(\mathbb{X}_2) = \boldsymbol{z_2}\right). \end{array} \right.$$

In order to compute the likelihoods and the predictions, the matrices $cov(Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_1)) = \sigma_1^2 \rho_{\theta_1}(\mathbb{X}_1, \mathbb{X}_1)$ and $cov(Z_2(\mathbb{X}_2), Z_2(\mathbb{X}_2)) = \sigma_2^2 \rho_{\theta_2}(\mathbb{X}_2, \mathbb{X}_2)$ must be invertible. If a positive definite covariance kernel is taken for Y_1, the fact that $\mathbb{X}_1$ is composed of distinct points sufficiently far from each other is sufficient to ensure the invertibility of the first matrix, as the matrix has distinct rows in this case. Because of the nullity property of Z_2: $Z_2(x_1, 0.7x_1 + 0.2) = 0$ (for all x_1 in $[0, 1]$), its covariance kernel is not positive definite, therefore the same constraint on $\mathbb{X}_2$ is not sufficient. An additional constraint must be added, that the points of $\mathbb{X}_2$ are sufficiently far from the nullity subspace $\mathcal{D} = \{(x_1, 0.7x_1 + 0.2), x_1 \in [0, 1]\}$, i.e. for all k in $[\![1, n_2]\!]$, $x_2^{(i_k)}$ must be sufficiently far from $0.7x_1^{(i_k)} + 0.2$. This second constraint ensures that the covariance matrix has no rows full of zeros.

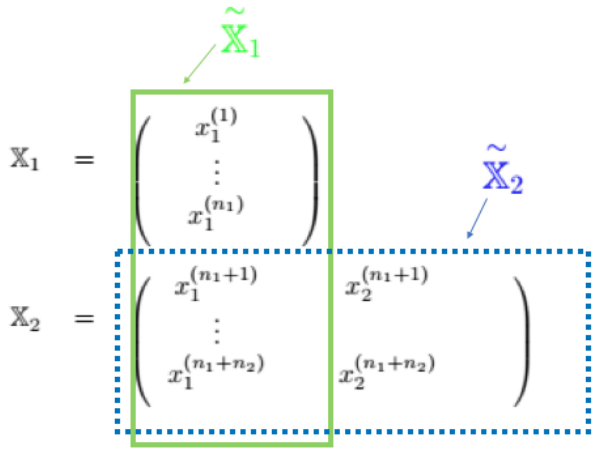
Non-nested designs The designs $\mathbb{X}_1$ and $\mathbb{X}_2$ are said non-nested if they have no common values of x_1, i.e. if they are of the form:

$$\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}, \ \mathbb{X}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} \\ \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} \end{pmatrix}, \ \{x_1^{(1)}, \ldots, x_1^{(n_1)}\} \cap \{x_1^{(n_1+1)}, \ldots, x_1^{(n_1+n_2)}\} = \emptyset.$$

An example of such designs is shown in figure below:



The prediction cannot be simplified but the estimation can. To be reduced to the previous case with nested designs, new designs must be defined, that take the role previously played by $\mathbb{X}_1$ and $\mathbb{X}_2$. These new designs, denoted $\widetilde{\mathbb{X}}_1$ and $\widetilde{\mathbb{X}}_2$, are defined as: $\widetilde{\mathbb{X}}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1+n_2)} \end{pmatrix}$, $\widetilde{\mathbb{X}}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} \\ \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} \end{pmatrix} = \mathbb{X}_2$. $\widetilde{\mathbb{X}}_1$ contains all the x_1 values of the matrices $\mathbb{X}_1$ and $\mathbb{X}_2$. $\widetilde{\mathbb{X}}_2$ contains all the (x_1, x_2) values of $\mathbb{X}_2$ so is equal to $\mathbb{X}_2$. The building of $\widetilde{\mathbb{X}}_1$ and $\widetilde{\mathbb{X}}_2$ is illustrated in figure below:



These new designs are nested: $\widetilde{\mathbb{X}}_2 \sqsubset \widetilde{\mathbb{X}}_1$ (see definition 7 in subsection 4.1.1) and thus, the complete data loss function (i.e. associated to $(Y_1(\widetilde{\mathbb{X}}_1), Z_2(\widetilde{\mathbb{X}}_2))$, which can be decoupled, is more adapted than the loss function associated with $(Y_1(\mathbb{X}_1), Y_2(\mathbb{X}_2))$, which cannot. The problem is that the value of Y_1 is not known everywhere in $\widetilde{\mathbb{X}}_1$, so some data is missing to use the complete data loss function. The EM algorithm is then used, in which the complete data loss function is replaced by its expectation conditioned by the training data:

$$\begin{aligned} \mathcal{Q}(\eta, \eta^*) &= \mathbb{E}_{\eta^*}\left[l_1(\eta_1; Y_1(\widetilde{\mathbb{X}}_1)) + l_2(\eta_2; Z_2(\widetilde{\mathbb{X}}_2)) \mid \boldsymbol{Y} = \boldsymbol{y} \right] \\ &= \underbrace{\mathbb{E}_{\eta^*}\left[l_1(\eta_1; Y_1(\widetilde{\mathbb{X}}_1)) \mid \boldsymbol{Y} = \boldsymbol{y} \right]}_{=\mathcal{Q}_1(\eta_1, \eta^*)} + \underbrace{\mathbb{E}_{\eta^*}\left[l_2(\eta_2; Z_2(\widetilde{\mathbb{X}}_2)) \mid \boldsymbol{Y} = \boldsymbol{y} \right]}_{=\mathcal{Q}_2(\eta_2, \eta^*)}. \end{aligned}$$

The EM algorithm consists in computing a sequence $(\eta^{(i)})_{i \geqslant 0}$ in which $\eta^{(i+1)}$ is built from $\eta^{(i)}$ by minimizing $\mathcal{Q}(\eta, \eta^{(i)})$. The subvectors $\eta_1^{(i+1)}$ and $\eta_2^{(i+1)}$ are built separately by minimizing respectively $\mathcal{Q}_1(\eta_1, \eta^{(i)})$ and $\mathcal{Q}_2(\eta_2, \eta^{(i)})$.

To optimize $\mathcal{Q}_1$ and $\mathcal{Q}_2$, the matrices $cov\left(Y_1(\widetilde{\mathbb{X}}_1), Y_1(\widetilde{\mathbb{X}}_1)\right) = \sigma_1^2 \rho_{\theta_1}(\widetilde{\mathbb{X}}_1, \widetilde{\mathbb{X}}_1)$ and $cov\left(Z_2(\widetilde{\mathbb{X}}_2), Z_2(\widetilde{\mathbb{X}}_2)\right) = \sigma_2^2 \rho_{\theta_2}(\widetilde{\mathbb{X}}_2, \widetilde{\mathbb{X}}_2)$ need to be inverted. As for the nested case, $\widetilde{\mathbb{X}}_1$ must have distinct points sufficiently far from each other, and $\widetilde{\mathbb{X}}_2$ must have distinct points sufficiently far from each other and sufficiently far from the nullity subspace $x_2 = 0.7x_1 + 0.2$. The last constraint means that, for all k in $[\![1, n_2]\!]$, $x_2^{(n_1+k)}$ must be sufficiently far from $0.7x_1^{(n_1+k)} + 0.2$.

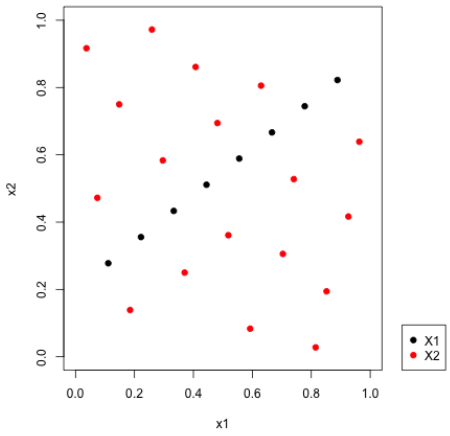


Figure 4.1: Example of training samples used for the test of the EM estimation.

	$m = 0$		$\sigma_1^2 = 1$		$\theta_1 = 0.5$		$\sigma_2^2 = 1$		$\theta_{21} = 0.5$		$\theta_{22} = 0.5$	
	Red	P	Red	P	Red	P	Red	P	Red	P	Red	P
Median	−0.18	**−0.10**	0.31	**0.4**	0.41	**0.42**	**1.00**	1.04	**0.50**	0.54	**0.51**	0.53
Interquartile range	0.94	**0.79**	**0.62**	0.95	**0.20**	0.23	**1.07**	1.22	0.19	**0.17**	**0.22**	0.24

Table 4.1: Estimations of the seqGPR parameters in case Z_2 is a P or Red process. Results are shown in terms of median and interquartile range of estimation. The best results are in bold.

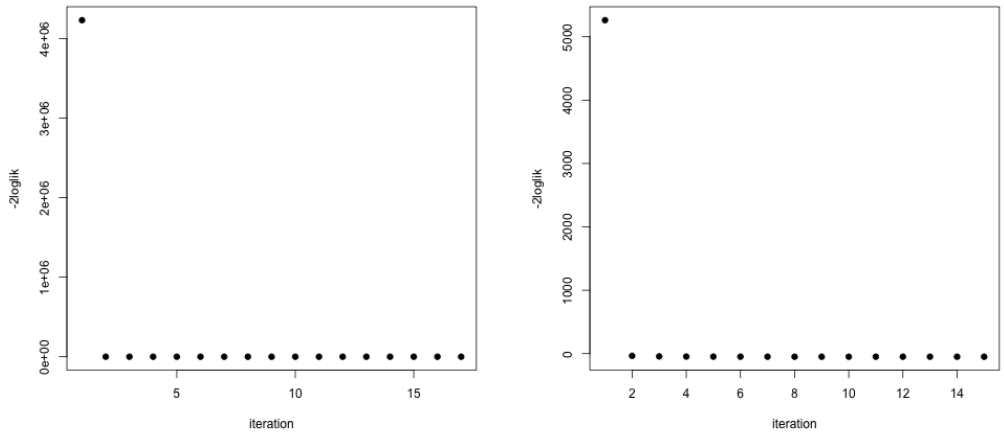
Application The EM algorithm is tested in the context of the example in 2D from subsection 3.1.2 in chapter 3, where the process $Y_2(x_1, x_2) = Y_1(x_1) + Z_2(x_1, x_2)$, with $Y_1(x_1) = m + Z_1(x_1)$ and $Z_2(x_1, 0.7x_1 + 0.2) = 0$, is simulated. The mean of the process is equal to $m = 1$. Z_1 (resp. Z_2) has a variance parameter equal to $\sigma_1^2 = 1$ (resp. equal to $\sigma_2^2 = 1$) and a covariance parameter equal to $\theta_1 = 0.5$ (resp. equal to $\theta_2 = (0.5, 0.5)$). The simulation of the process Y_2 is realised 100 times, each time on a DoE $\mathbb{X}_1$ of size 8 and a DoE $\mathbb{X}_2$ of size 18 (see figure 4.1). $\mathbb{X}_1$ and $\mathbb{X}_2$ are non-nested, $\mathbb{X}_2$ is built following the method described in section 5.2 in chapter 5.

For both Red and P the EM algorithm seems to have converged towards a local optimum of the likelihood as shown for one particular DoE in figure 4.2.

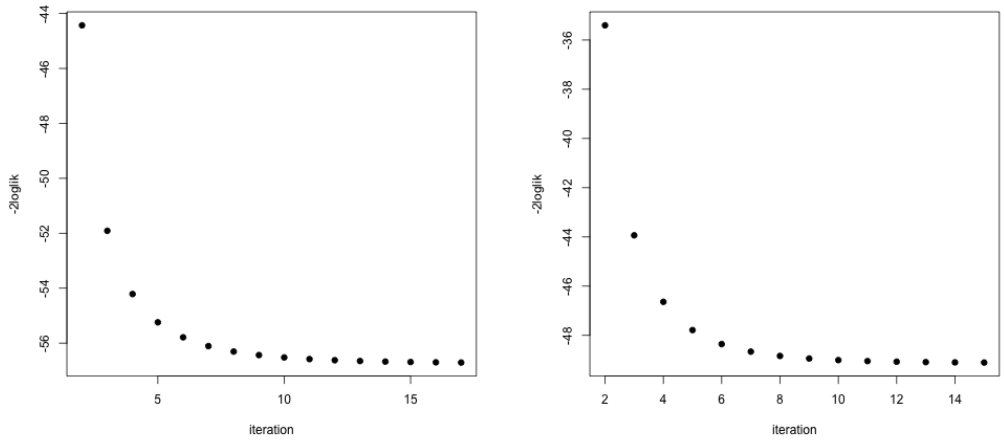
The comparison of the estimations in the case Z_2 is a Red or P process is shown in table 4.1. The estimations are good for the covariance parameters (θ_1 and θ_2) and less precise for the mean and variance parameters. Red and P are globally equivalent in terms of parameter estimation.

See appendix 9.1.2 for the example in 4D.

Now that the metamodel building method is complete, it must be evaluated.



(a) Red (b) P



(c) Red,zoom (d) P,zoom

Figure 4.2: Convergence of the loss function in the EM algorithm. Panels 4.2c and 4.2d show the values of the loss function starting from iteration 2 of the EM algorithm, whereas panels 4.2a and 4.2b show the values of the loss function starting from iteration 1.

4.2 Test cases

In what follows, three metamodels are compared on different test cases:

- seqGPR: the metamodel introduced in this thesis. The kernel of Y_1 and the ones on which are based $(Z_n)_{n=2}^N$ are stationary tensor product matern $\frac{5}{2}$ covariance kernels.

- K_N (N being the number of the last step): a classic kriging metamodel with a stationary tensor product matern $\frac{5}{2}$ covariance kernel and a constant mean which is built on the last training sample $(\mathbb{X}_N, \boldsymbol{y_N})$.

- K_tot: a classic kriging metamodel similar to K_N, but built on all training samples $(\mathbb{X}_1, \boldsymbol{y_1}), \cdots, (\mathbb{X}_N, \boldsymbol{y_N})$.

Four versions of seqGPR are compared. P_full and P_rob (resp. Red_full and Red_rob) use P (resp. Red) processes defined in section 3.2.3 (resp. 3.2.1) in chapter 3, respectively of type Full and Robust for the $(Z_n)_{n=2}^N$ (the types are developed in subsection 4.2.1). The metamodels are compared in terms of RMSE on a test sample. The RMSE formula is recalled below:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n_{test}} (y_{test}^{(i)} - \hat{y}(x_{test}^{(i)}))^2}{n_{test}}},$$

with $(x_{test}^{(i)})_{i=1}^{n_{test}}$ the test sample, $(y_{test}^{(i)})_{i=1}^{n_{test}}$ the observations on the test sample, and $(\hat{y}(x_{test}^{(i)}))_{i=1}^{n_{test}}$ the predictions of the metamodel on the test sample. A simple likelihood estimation is used for the two kriging models K_N and K_tot. An estimation by EM (in the non-nested case) or by independent maximization of the likelihoods of Y_1 and the $(Z_n)_{n=2}^N$ (in the nested case) is carried out for the metamodel seqGPR. Two test cases involve an analytic output and one describes an industrial case.

4.2.1 Robustness

The philosophy of the method seqGPR is to explain a maximum of the data by Y_1, and to correct it thanks to the $(Z_n)_{n=2}^N$, so all the parameters for Y_1 are estimated individually and on the contrary some parameters of the $(Z_n)_{n=2}^N$ are grouped, all parameters being equal in the same group. Two types of parametrization for the correction processes are defined:

- The first type is called Full. It is the usual one, where all components are estimated individually. The vector θ_n, of covariance parameters of Z_n, is equal to $\theta_n = (\theta_n^1, \cdots, \theta_n^{d_1+\cdots+d_n})$, with one component θ_n^i for each input variable x_i. This type of parametrization has the advantage of giving a great flexibility in the fitting to the training data, however it may lack robustness in the prediction on other data.

- The second type is called Robust. Some components are affected the same value and are estimated together while some others are estimated individually. The vector θ_n is equal to $\theta_n = (\underbrace{\alpha_n, \cdots, \alpha_n}_{I_1 \cup \cdots \cup I_{n-1}}, \underbrace{\theta_n^1, \cdots, \theta_n^{d_n}}_{I_n})$, with α_n common to all components of $x_{I_1 \cup \cdots \cup I_{n-1}}$, and one parameter θ_n^i for each component of x_{I_n}. This type is motivated by a search for compromise between bias and variance. Diminishing the number of parameters to estimate automatically decreases the chances of overfitting.

4.2.2 Analytic test case in dimension 4

The output f considered in this test case is a function of 4 inputs $x = (x_1, x_2, x_3, x_4)$: $f(x_1, x_2, x_3, x_4) = g_1(x_1, x_2) + g_2(x_1, x_2, x_3, x_4)$, with:

$$\begin{cases} g_1(x_1, x_2) & = & \left[4 - 2.1(4x_1 - 2)^2 + \frac{(4x_1-2)^4}{3}\right](4x_1 - 2)^2 \\ & + & (4x_1 - 2)(2x_2 - 1) + [-4 + 4(2x_2 - 1)^2](2x_2 - 1)^2, \\ g_2(x_1, x_2, x_3, x_4) & = & 4\exp\left(-\|x - 0.3\|^2\right). \end{cases}$$

g_1 is the six-hump camel back resized in $[0, 1]^2$. It is a heckled 2D surface. g_2 is a smoother Gaussian function of 4 inputs. The study is composed of two steps:

- At step 1, computer code evalutations are run at DoE $\mathbb{X}_1$ in dimension 2, such that $f_1(\mathbb{X}_1) = \boldsymbol{y_1}$, with f_1 defined by $f_1(x_1, x_2) = f(x_1, x_2, \grave{x}_3, \grave{x}_4)$. Only the first two variables x_1 and x_2 are free and the other two variables are fixed: $\grave{x}_3 = \frac{x_1+x_2}{2}$, $\grave{x}_4 = 0.2x_1 + 0.7$.

- At step 2, new simulations are launched at points $\mathbb{X}_2$, a design in dimension 4. The last two variables (x_3, x_4) are now released.

The total number of variables is $d = 4$, the number of steps is $N = 2$, the index set of variables released at step 1 is $I_1 = \{1, 2\}$ and the index set of variables released at step 2 is $I_2 = \{3, 4\}$.

Figure 4.3 shows RMSE of the different methods computed on a Sobol sequence of size 10000 used as a test set. The RMSE is computed for 100 different DoE's $(\mathbb{X}^1, \boldsymbol{y}^1)$ and $(\mathbb{X}^2, \boldsymbol{y}^2)$. The 100 RMSE's are represented by a boxplot for each metamodel. Different sizes of DoE's are tested. A comparison is made between nested designs (built following section 5.1 in chapter 5) and non-nested ones (built following section 5.2 in chapter 5). Different sizes of design are also compared to see the evolution of the RMSE with the number of training points and see the influence of the size of $\mathbb{X}_1$ relative to the size of $\mathbb{X}_2$:

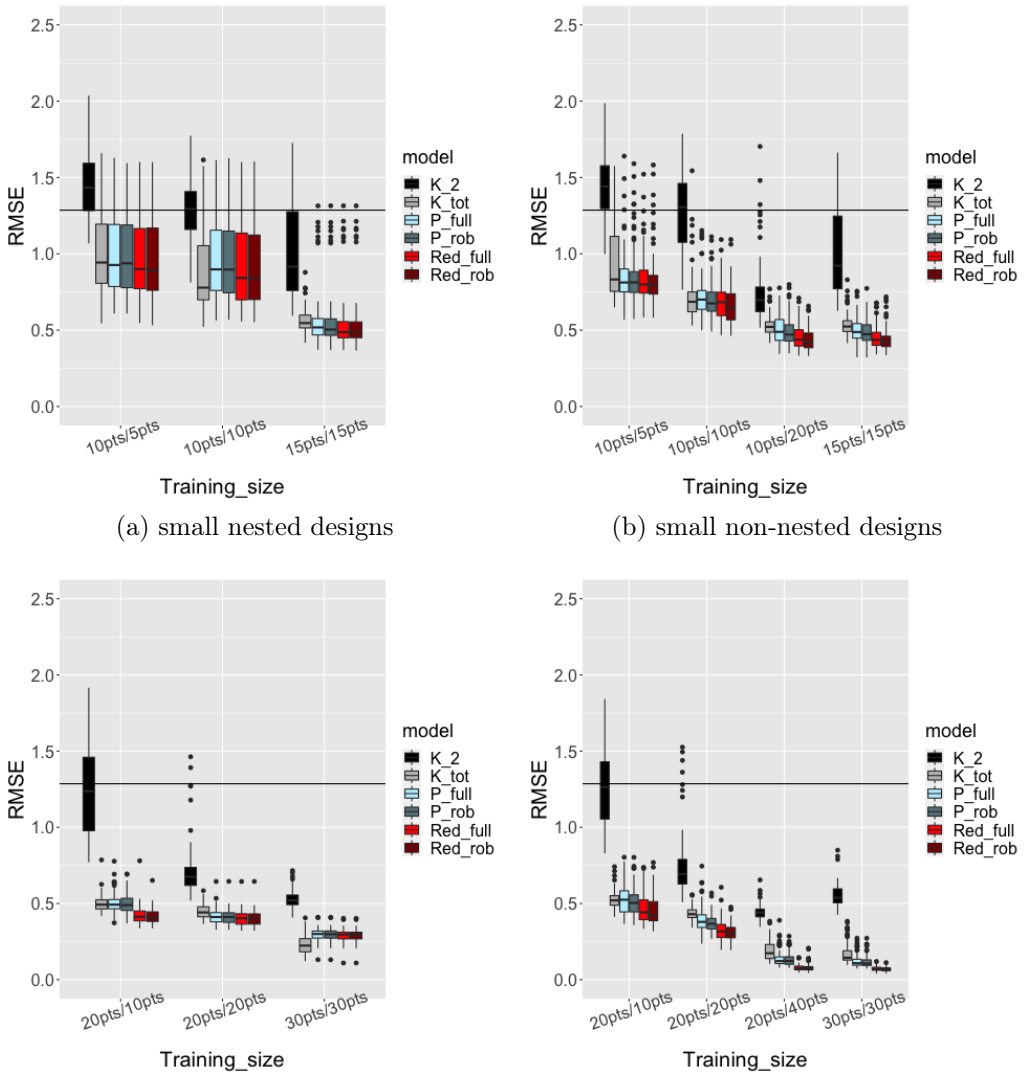
- $\mathbb{X}_1$ bigger than $\mathbb{X}_2$: 10pts/5pts ($\mathbb{X}_1$ of size 10 and $\mathbb{X}_2$ of size 5), 20pts/10pts

- $\mathbb{X}_1$ smaller than $\mathbb{X}_2$: 10pts/20pts, 20pts/40pts

- $\mathbb{X}_1$ of same size as $\mathbb{X}_2$: 10pts/10pts, 15pts/15pts, 20pts/20pts, 30pts/30pts.

The standard deviation of the output on the test set is represented by the black horizontal line. If the RMSE reaches this value, the corresponding metamodel is considered as not predictive.

As regards the influence of the training samples, all metamodel performances improve with the size of the training sample. Results show that including the previous information of $(\mathbb{X}^1, \boldsymbol{y}^1)$ is useful, as it greatly improves the performances of the metamodels. Indeed K_2 is by far the worst metamodel and sometimes is even not predictive, as it reaches the black line. The metamodels built on non-nested designs are often singularly more accurate than the metamodels built on nested designs. In the case of non-nested designs, which is the most interesting, no significant difference is shown in the distribution of training points between $\mathbb{X}_1$ and $\mathbb{X}_2$: the case with 10 points for $\mathbb{X}_1$ and 20 points for $\mathbb{X}_2$, or with 15 points for $\mathbb{X}_1$ and 15 points for $\mathbb{X}_2$ give similar accuracy. The same conclusion can be drawn from the



(a) small nested designs

(b) small non-nested designs

(c) big nested designs

(d) big non-nested designs

Figure 4.3: RMSE's (see equation 2.2) of the different metamodels on 100 studies. The black line is the standard deviation of the output on the test sample. In panels 4.3a and 4.3b, small training sizes are tested : 10pts/5pts ($\mathbb{X}_1$ of size 10, $\mathbb{X}_2$ of size 5), 10pts/10pts ($\mathbb{X}_1$ of size 10, $\mathbb{X}_2$ of size 10), 10pts/20pts ($\mathbb{X}_1$ pf size 10, $\mathbb{X}_2$ of size 20), and 15pts/15pts ($\mathbb{X}_1$ of size 15, $\mathbb{X}_2$ of size 15). In panels 4.3c and 4.3d, bigger training sizes are tested : 20pts/10pts ($\mathbb{X}_1$ of size 20, $\mathbb{X}_2$ of size 10), 20pts/20pts ($\mathbb{X}_1$ of size 20, $\mathbb{X}_2$ of size 20), 20pts/40pts ($\mathbb{X}_1$ of size 20, $\mathbb{X}_2$ of size 40), and 30pts/30pts ($\mathbb{X}_1$ of size 30, $\mathbb{X}_2$ of size 30). In panels 4.3a and 4.3c, the designs are nested whereas in panels 4.3b and 4.3d, the designs are non-nested.

comparison of the case with 20 points for $\mathbb{X}_1$ and 40 points for $\mathbb{X}_2$, or with 30 points for $\mathbb{X}_1$ and 30 points for $\mathbb{X}_2$.
As regards the comparison of the metamodels for the non-nested design case, the robust seqGPR metamodels are better than the full, they are equivalent to or better than K_tot. It seems that the cases with a small or high number of training points are more discriminating than the middle cases. With a small number of points (10pts/5pts), K_tot seems more destabilized than seqGPR. With a high number of points (20pts/40pts), seqGPR is more accurate than K_tot. Finally, seqGPR using Red is better than seqGPR using P.

In the following test cases, only the robust versions of seqGPR, which are more performing and whose parameter estimation is less complex, are compared to K_tot, and only the non-nested designs are used.

4.2.3 Analytic test case in dimension 15

The following test case should be less favorable to seqGPR. The objective function considered is in higher dimension and not decomposed as a sum of functions that respects the order in which the variables are released

$$f \quad : \left| \begin{array}{lll} [-3,3]^{15} & \to & \mathbb{R} \\ x & \mapsto & a_1' x + a_2' \sin x + a_3' \cos x + x' M x. \end{array} \right. \tag{4.5}$$

The function f was first used in [Oakley and O'Hagan, 2004]. The values of its coefficients a_1, a_2, a_3 and M can be found in www.sheffield.ac.uk/st1jeo. The inputs are rescaled in $[0,1]$ and are rearranged by decreasing order of Sobol index, but for the sake of simplicity, the resulting output is still denoted by f.

In practice, the first studies are done on the most influential inputs, chosen according to experts' physical knowledge. Here, as the function is analytical, the choice of the releasing order for the inputs is made by sensitivity analysis, which is directly performed on the function. Following the distribution of the Sobol indices (see subsection 2.1.4 in chapter 2 for their definition) shown on figure 4.4, a study of $N = 3$ steps is made, adding respectively the groups of variables $I_1 = \{1,2,3\}$, $I_2 = \{4,5,6,7,8,9\}$, and $I_3 = \{10,11,12,13,14,15\}$. The variables that are fixed are set to 0.5. Three non-nested designs (built following the method suggested in section 5.2 in chapter 5), one for each step, are generated with 5 points per dimension considered at the step: $\mathbb{X}_1 \subset [0,1]^3$ of size 15, $\mathbb{X}_2 \subset [0,1]^9$ of size 45, and $\mathbb{X}_3 \subset [0,1]^{15}$ of size 75. A Sobol sequence of size 10000 in dimension 15 is used as a test sample. The study is done 100 times, each time with new DoE's.

The performances of the metamodels are shown in table 4.2. All versions of seqGPR are better than the classic kriging, with a slight improvement for the Red version in comparison with the P version. Red and P are almost equivalent as opposed to the previous test case. This can be due to the fact that the P version is this time based on a stationary latent process contrarily to the previous test case. Indeed the variables which are fixed at a given step are set to a constant instead of varying in function of the free variables, as it is the case in the previous test case.

4.2.4 Industrial test case

The industrial product which is studied in this section is the fan system (presented in section 1.1 in chapter 1) which is part of a car engine cooling system from Valeo company (see figure

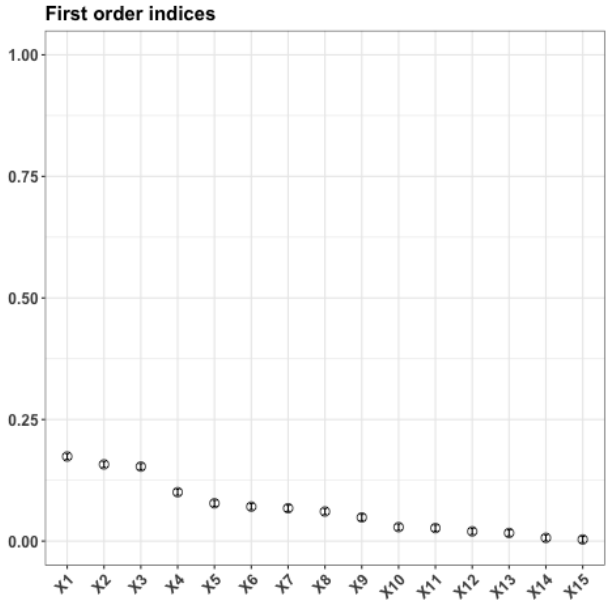
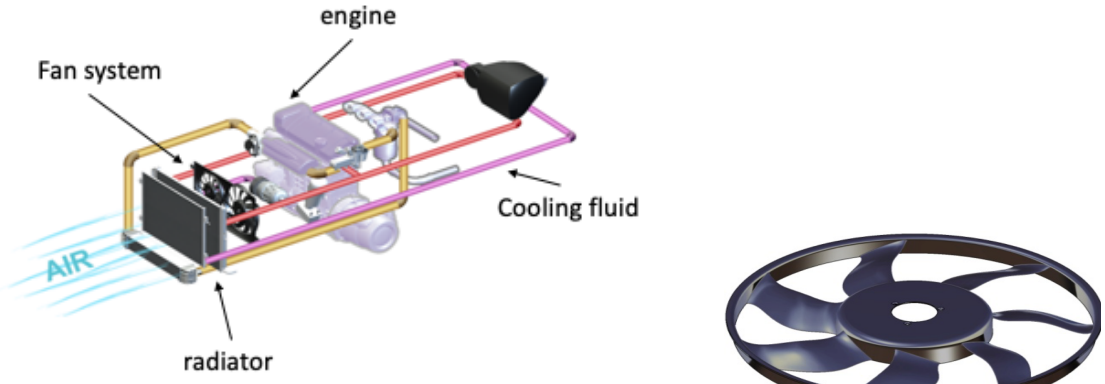


Figure 4.4: Sobol indices of the objective function (4.5).

	K_tot	P_rob	Red_rob
Median ($\cdot 10^{-4}$)	46.039	37.135	**36.903**
Interquartile range ($\cdot 10^{-4}$)	7.258	6.660	**6.570**

Table 4.2: Performance of the metamodels for the 100 studies made on the analytic 15D test case. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$). The best performances are in bold.



(a) Car engine cooling system

(b) Fan system

Figure 4.5: On left panel, diagram of the car engine cooling system. On right panel, diagram of fan system. [Valeo,]

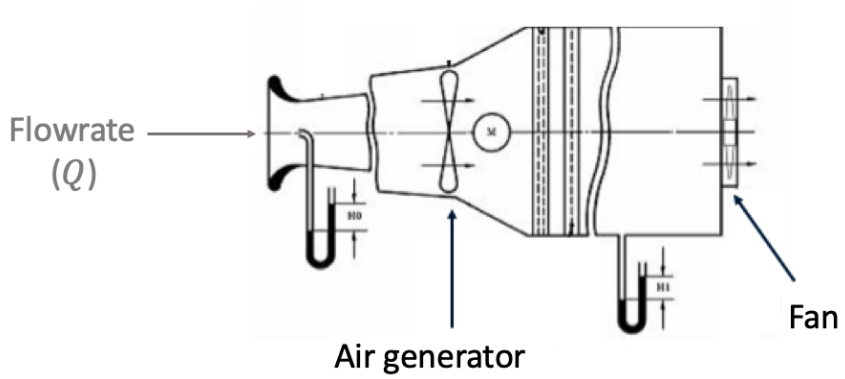
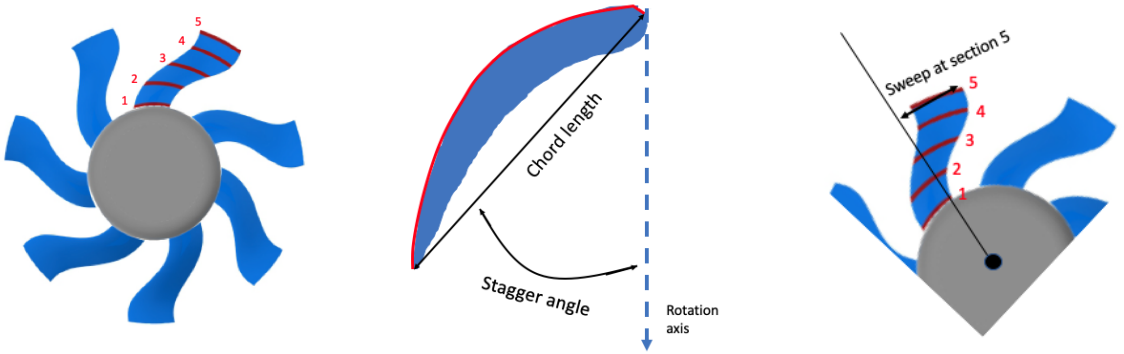


Figure 4.6: Description of the environment chosen to model the fan system in the computer code. The flowrate is imposed by an air generator. [Valeo,]

4.5). As a recall, the car engine is cooled by a fluid. The temperature of this fluid is then decreased by passing through a heat exchanger, which constitutes an interface between the fluid and a wind projected on it. The wind can be generated naturally by the car speed, or by fans which are activated when the car speed is not sufficient.

The fan is modeled outside the engine cooling system (see figure 4.6). The output considered here is the Pressure difference (ΔP) between the upstream and the downstream of the air flow crossing the fan. 15 variables are kept as input of the model. The first input of the model is the flowrate (see figure 4.6) which is not generated by the fan but by an air generator. The 14 others are geometric: 5 stagger angles and chord lengths corresponding to the five sections monitoring the geometry of the blade, and 4 sweeps for sections 2 to 4 (see figure 4.7). The chord is the line joining the two borders of the blade at the considered section. The stagger angle is the inclination angle of the chord. The sweep is the distance between the black line, intersecting the rotation axis and the leading edge (right border on the figure) at the first section, and the leading edge at the considered section (see figure 4.7c).
To summarize, the variables are

- The flowrate: Q



(a) Sections

(b) Chord length and stagger angle at a given section

(c) Definition of the sweep at section 5

Figure 4.7: On the left panel, visualization of the five sections of the blade. On the middle panel, definition of the stagger angle and the chord length at a given section. On the right panel, definition of the sweep at section 5.

- The stagger angles at the five sections: $Stag1$, $Stag2$, $Stag3$, $Stag4$, $Stag5$
- The chord lengths at the five sections: $Chord1$, $Chord2$, $Chord3$, $Chord4$, $Chord5$
- The sweeps at sections 2 to 5: $Swe2$, $Swe3$, $Swe4$, $Swe5$

In the rest of this test case, all variables are adimensionalized in $[0, 1]$ and the output ΔP is adimensionalized in $[-1, 1]$.

Industrial experts at Valeo have carried out a two-step ($N = 2$) study in the context of this work.

- At step 1, all the sweeps are fixed to constants: $Swe2 = 0.517645$, $Swe3 = 0.82$, $Swe4 = 1$, $Swe5 = 1$. An OLH (see subsection 2.1.5 in chapter 2 for its definition) of size 126 has been created on the 11 free inputs ($I_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$).
- At step 2, all the inputs are free ($I_2 = \{12, 13, 14, 15\}$). An OLH of size 299 has been created for the 15 inputs.

The seqGPR metamodel is fitted on $\mathbb{X}_1$ and $\mathbb{X}_2$ where $\mathbb{X}_1$ (resp. $\mathbb{X}_2$) is a subset of size 50 extracted from the OLH generated at step 1 (resp. a subset of size 50 extracted from the OLH generated at step 2). The two subsets $\mathbb{X}_1$ and $\mathbb{X}_2$ are optimized for the generalized maximin criterion, they are non-nested. The complementary subset of $\mathbb{X}_2$ is used as a test sample to evaluate the method. The procedure is made 30 times, each time new $\mathbb{X}_1$ and $\mathbb{X}_2$ are extracted. This cross-validation methodology enables to have a robust comparison of the models, and to test them in an extreme case with very few training points.

Table 4.3 shows performances of the different metamodels on the test samples. The metamodel seqGPR is better than K_tot. Its version with P is this time slightly better than with Red, again probably because the fixed inputs are constant as in the previous test case.

	K_tot	P_rob	Red_rob
Median ($\cdot 10^{-3}$)	44.585	**41.207**	42.088
Interquartile range ($\cdot 10^{-3}$)	5.892	4.657	**4.609**

Table 4.3: Performance of the metamodels for the 100 studies made on the industrial 15D test case. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$). The best performances are in bold.

4.3 Conclusion

In this chapter, two estimation methods are suggested. The first one is a decoupling of the likelihoods in case the designs are nested. The parameters of the Z_i are estimated individually by optimizing the likelihoods of the Z_i whose formulae are written in 3.4. The use of nested designs has the advantage of decoupling the predictions too. Their drawbacks are that they limit the size of the designs and the exploration of the input space. The second method is an EM algorithm that uses this decoupling of the likelihood in a sequential way. It uses non-nested designs that have the drawback of not decoupling the likelihood and prediction anymore. However, their size is not limited and neither is the exploration of the input space. This new estimation method, which transforms one complex optimization problem in several easier ones, is tested on a simulated example and the estimations are compared between Red and P processes. The conclusion is that they are equivalent and that the estimation is good for the covariance parameters and less accurate for the variance and mean parameters as it is the case for classic kriging.

The prediction accuracy of different versions of the metamodel seqGPR is compared to a classic kriging metamodel on one simple analytic test case, one more complicated and the industrial test case which motivated the thesis. The first conclusion is that the previous training samples, generated before the current step, bring useful information and improve the prediction of the metamodels. The second conclusion is that the non-nested DoE's give better results than the nested ones. Finally, the method seqGPR is competitive, and especially its Red_robust version (using Red process for the $(Z_n)_{n=2}^N$ with robust parametrization of their kernel).

Chapter 5

Designs of Experiments

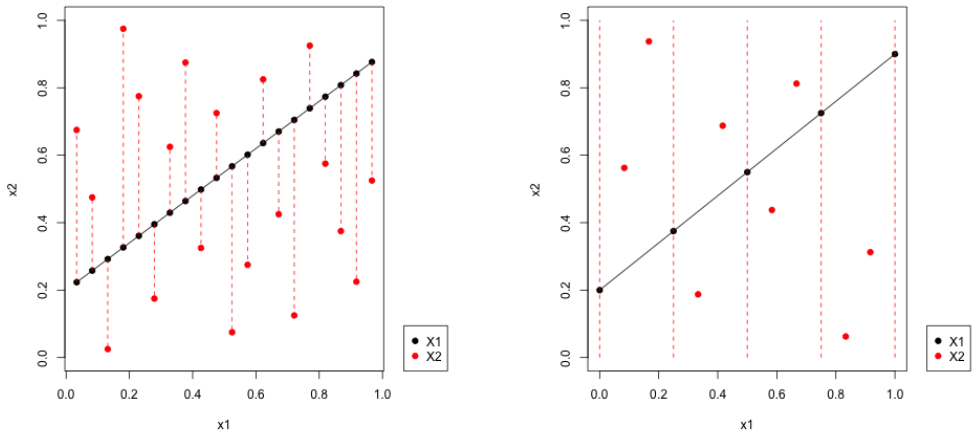
This chapter describes building methods for the DoE's $\mathbb{X}_1, \cdots, \mathbb{X}_N$ in the seqGPR metamodel context (see equation (3.1) in chapter 3). As ρ_1 is a usual positive-definite kernel used in kriging, the only source of singularity due to $\mathbb{X}_1$ for the covariance matrices involved in the estimation process is the presence of identical rows caused by redundant points in $\mathbb{X}_1$. A good value of the generalized maximin criterion (see subsection 2.1.5 in chapter 2) is sufficient to ensure a good space filling property of $\mathbb{X}_1$ in $[0,1]^{d_1}$ and the good conditioning of $\rho_1(\mathbb{X}_1, \mathbb{X}_1)$ (as the points are far from each other). Therefore, $\mathbb{X}_1$ is taken as an LHS optimized for the generalized maximin criterion. However, the $(\mathbb{X}_n)_{n=2}^N$ must verify other properties. The building methods introduced below enable a good space-filling of the samples and the invertibility of the covariance matrices involved in the estimation process (see chapter 4). It is assumed that the training sample sizes n_n of $\mathbb{X}_n$ ($n \in [\![1, N]\!]$) are imposed, that the designs are built recursively and greedily (first $\mathbb{X}_1$, then $\mathbb{X}_2$, $\cdots$, then $\mathbb{X}_N$), and that the building method is identical for all designs from $\mathbb{X}_2$ to $\mathbb{X}_N$.

A building method of $(\mathbb{X}_n)_{n\in[\![2,N]\!]}$ must be defined in a nested designs context. These designs are interesting as they enable the decoupling of the loss function for the parameter estimation (see subsection 4.1.1 in chapter 4). However, they are very constrained on the size which cannot increase with the step, and on the input space exploration, which is limited. In the case of nested designs, the covariance matrices that need to be inverted in the likelihoods are the $(cov(Z_n(\mathbb{X}_n), Z_n(\mathbb{X}_n)))_{n=1}^N$. A building method must also be suggested to generate non-nested samples for the seqGPR metamodel. Non-nested designs have the advantage of being more flexible: no constraint on the size and more possibility to explore the input space. However the parameter estimation is more complicated as the loss function is not decoupled. The covariance matrices that need to be inverted in the EM algorithm (see subsection 4.1.2 in chapter 4) are the $\left(cov(Z_n(\widetilde{\mathbb{X}}_n), Z_n(\widetilde{\mathbb{X}}_n))\right)_{n=1}^N$. Nested and non-nested designs are illustrated on figure 5.1 in the case of the example in 2D of subsection 3.1.2 in chapter 3.

Building methods are described for samples $\mathbb{X}_2$ to $\mathbb{X}_N$, in the case of nested designs in section 5.1, and in the case of non-nested designs in section 5.2.

5.1 Nested designs

This section describes the building method for nested designs. The constraints on $\mathbb{X}_n$ ($n \geqslant 2$), necessary to invert $cov(Z_n(\mathbb{X}_n), Z_n(\mathbb{X}_n))$, are enumerated and lead to an optimization problem. An algorithm is then implemented to solve this optimization problem. As a recall, the size of the design $\mathbb{X}_n$ cannot increase with n: $n_N \leqslant n_{N-1} \leqslant \cdots \leqslant n_1$.



(a) Nested designs

(b) Non-nested designs

Figure 5.1: Illustration of nested (panel 5.1a) and non-nested (panel 5.1b) designs for the 2D example of subsection 3.1.2 in chapter 3. $\mathbb{X}_1$ (resp. $\mathbb{X}_2$) is represented by black points (resp. red points). The nullity subspace $x_2 = 0.7x_1 + 0.2$ is represented by the black line.

5.1.1 Iterative construction procedure

In addition to the space-fillingness and remoteness of the points, $\mathbb{X}_n$ must be nested in $\mathbb{X}_{n-1}$ ($\mathbb{X}_n \sqsubset \mathbb{X}_{n-1}$, see definition 7 in subsection 4.1.1 in chapter 4), that means that it must share the same values for the columns corresponding to the variables $x_{I_1 \cup \cdots \cup I_{n-1}}$. Secondly, because of the nullity property (see equation (3.3)) of Z_n, a new source of singularity appears which is the presence of rows full of zeros at points where Z_n is null almost surely. $\mathbb{X}_n$ must then have an additional property which is its remoteness from the subspace $\{(x_{I_1 \cup \cdots \cup I_{n-1}}, \grave{x}_{I_n}),\ x_{I_1 \cup \cdots \cup I_{n-1}} \in [0,1]^{d_1 + \cdots + d_{n-1}}\}$.

The following optimization problem is chosen to build $\mathbb{X}_n$:

$$\begin{cases} \min\limits_{n \subset [0,1]^{d_1 + \cdots + d_n}} \Phi_n(\mathbb{X}_n), \\ \\ \text{under the constraint: } \mathbb{X}_n \sqsubset \mathbb{X}_{n-1}, \end{cases}$$

with Φ_n derived from the generalized maximin criterion (see subsection 2.1.5 in chapter 2):

$$\Phi_n(\mathbb{X}_n) = \underbrace{\left(\sum_{\substack{x,t \in \mathbb{X}_n \\ x \neq t}} \left(\frac{1}{\|x - t\|^2_{[0,1]^{d_1 + \cdots + d_n}}} \right)^p \right)^{\frac{1}{p}}}_{=\phi_1} + \underbrace{\left(\sum_{x \in \mathbb{X}_n} \left(\frac{1}{\|x_{I_n} - \grave{x}_{I_n}\|^2_{[0,1]^{d_n}}} \right)^p \right)^{\frac{1}{p}}}_{\phi_2}.$$

ϕ_1 ensures the space-fillingness and remoteness of the points of $\mathbb{X}_n$ and ϕ_2 ensures that $\mathbb{X}_n$ is far from the subspace where Z_n is null. The default value of p is 50 in the R package DiceDesign and this value is imposed in what follows.

5.1.2 Numerical implementation

The algorithm consists in interverting rows from an initial design $\mathbb{X}^{(0)}$ by a simulated annealing process. Denoting the sample $\mathbb{X}_{n-1}$ by:

$$\mathbb{X}_{n-1} = \begin{pmatrix} x^{(1)}_{I_1 \cup \cdots \cup I_{n-1}} \\ \vdots \\ x^{(n_{n-1})}_{I_1 \cup \cdots \cup I_{n-1}} \end{pmatrix},$$

the initial design is taken equal to:

$$\mathbb{X}^{(0)} = \left[\begin{array}{c|ccc} x^{(1)}_{I_1 \cup \cdots \cup I_{n-1}} & \frac{1}{2n_n} & \cdots & \frac{1}{2n_n} \\ \vdots & \vdots & & \vdots \\ x^{(k)}_{I_1 \cup \cdots \cup I_{n-1}} & \frac{1}{2n_n} + \frac{k-1}{n_n} & \cdots & \frac{1}{2n_n} + \frac{k-1}{n_n} \\ \vdots & \vdots & & \vdots \\ x^{(n_n)}_{I_1 \cup \cdots \cup I_{n-1}} & 1 - \frac{1}{2n_n} & \cdots & 1 - \frac{1}{2n_n} \\ \hline x^{(n_n+1)}_{I_1 \cup \cdots \cup I_{n-1}} & -1 & \cdots & -1 \\ \vdots & \vdots & & \vdots \\ x^{(n_{n-1})}_{I_1 \cup \cdots \cup I_{n-1}} & -1 & \cdots & -1 \end{array}\right]$$

The leftmost columns correspond to the already free variables at step $n-1$, $x_{I_1 \cup \cdots \cup I_{n-1}}$. In the simulated annealing algorithm, all the rows can be interverted from row 1 to row n_{n-1} for these columns. To keep the nesting property, the components on the same row are constrained to stay together, i.e. only interversions between entire $x^{(i)}_{I_1 \cup \cdots \cup I_{n-1}}$ are allowed. The rightmost columns correspond to the newly released variables at step n, x_{I_n}. Their values in the design belong to the dicretization $\{\frac{1}{2n_n} + \frac{k-1}{n_n},\ k \in [\![1, n_n]\!]\}$. In the simulated annealing algorithm, only rows from 1 to n_n can be interverted, so that the "-1" stay in the last rows. The "-1" are just added to complete the lines that are not kept. The initial associated candidate sample is :

$$\mathbb{X}^{(0)}_c = \left[\begin{array}{c|ccc} x^{(1)}_{I_1 \cup \cdots \cup I_{n-1}} & \frac{1}{2n_n} & \cdots & \frac{1}{2n_n} \\ \vdots & \vdots & & \vdots \\ x^{(k)}_{I_1 \cup \cdots \cup I_{n-1}} & \frac{1}{2n_n} + \frac{k-1}{n_n} & \cdots & \frac{1}{2n_n} + \frac{k-1}{n_n} \\ \vdots & \vdots & & \vdots \\ x^{(n_n)}_{I_1 \cup \cdots \cup I_{n-1}} & 1 - \frac{1}{2n_n} & \cdots & 1 - \frac{1}{2n_n} \end{array}\right].$$

The simulated annealing algorithm is shown in algorithm 1 in appendix 9.4.2. It generates a sequence $\left(\mathbb{X}^{(l)}\right)_{l \geqslant 0}$ from which the corresponding sequence of candidate samples $\left(\mathbb{X}^{(l)}_c\right)_{l \geqslant 0}$ is deduced by removing the rows of $\mathbb{X}^{(l)}$ containing components equal to -1. $\mathbb{X}_n$ is then equal to the last term of the sequence $\left(\mathbb{X}^{(l)}_c\right)_{l \geqslant 0}$, when the maximum of iterations or the threshold temperature has been reached.

5.1.3 Example in 2D

In the example in 2D from subsection 3.1.2 in chapter 3, in order for $\rho_2(\mathbb{X}_2, \mathbb{X}_2)$ to be well-conditioned, $\mathbb{X}_2$ must have distinct points sufficiently far from each other (to avoid identical

rows) and from the nullity subset of Z_2 (to avoid rows full of zeros): $\mathcal{D} = \{(x_1, 0.7x_1 + 0.2), x_1 \in [0,1]\}$. The last constraint means that: for all k in $[\![1, n_2]\!]$, $x_2^{(i_k)}$ must be far from $0.7x_1^{(i_k)} + 0.2$.

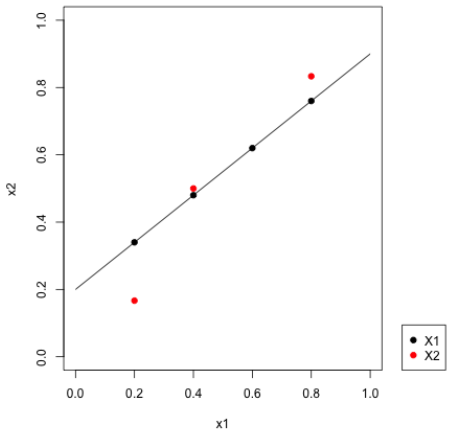
The following optimization problem is chosen to built $\mathbb{X}_2$:

$$\begin{cases} \min\limits_{\mathbb{X}_2 \subset [0,1]^2} \Phi_2(\mathbb{X}_2), \\ \\ \text{under the constraint: } \mathbb{X}_2 \sqsubset \mathbb{X}_1, \end{cases}$$

with Φ_2 derived from the generalized maximin criterion (see subsection 2.1.5 in chapter 2):

$$\begin{aligned} \Phi_2(\mathbb{X}_2) &= \underbrace{\left(\sum_{k=1}^{n_2} \sum_{\substack{l=1 \\ l \neq k}}^{n_2} \left(\frac{1}{(x_1^{(i_k)} - x_1^{(i_l)})^2 + (x_2^{(i_k)} - x_2^{(i_l)})^2} \right)^{50} \right)^{\frac{1}{50}}}_{=\phi_1} \\ &+ \underbrace{\left(\sum_{k=1}^{n_2} \left(\frac{1}{(x_2^{(i_k)} - 0.7x_1^{(i_k)} - 0.2)^2} \right)^{50} \right)^{\frac{1}{50}}}_{=\phi_2}. \end{aligned}$$

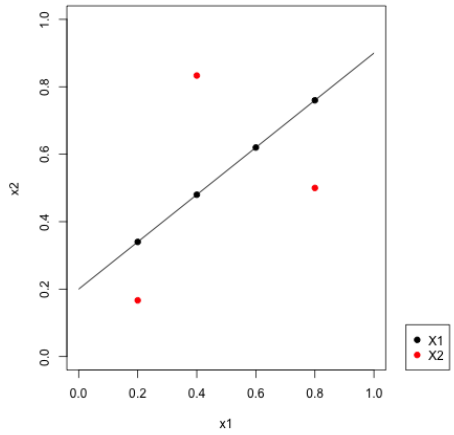
To illustrate the building algorithm, it is assumed that nested samples $\mathbb{X}_1$, $\mathbb{X}_2$, respectively of size 4, and 3, must be generated. At step 1, the training sample is supposed to be equal to $\mathbb{X}_1 = \begin{pmatrix} 0.2 \\ 0.4 \\ 0.6 \\ 0.8 \end{pmatrix}$. The variable x_2 is fixed to $0.7x_1 + 0.2$. At step 2, the goal is to generate a sample $\mathbb{X}_2$ nested in $\mathbb{X}_1$. $\mathbb{X}_2$ contains the values of x_1, x_2. The nesting property of $\mathbb{X}_2$ implies that its size is smaller than or equal to the size of $\mathbb{X}_1$. For example, it is here imposed to 3. To build $\mathbb{X}_2$, a matrix is created containing the first column of $\mathbb{X}_1$ and initializing the values of x_2 in the second column as a discretization of $[0,1]$: $\begin{pmatrix} 0.2 & \frac{1}{6} \\ 0.4 & \frac{3}{6} \\ 0.6 & \frac{5}{6} \\ 0.8 & -1 \end{pmatrix}$. The matrix $\mathbb{X}_2$ is deduced by taking the rows where $x_2 \neq -1$. The initial value of $\mathbb{X}_2$ is illustrated in figure below.



$$\mathbb{X}_2 \rightarrow \begin{pmatrix} 0.2 & \frac{1}{6} \\ 0.4 & \frac{3}{6} \\ 0.6 & \frac{5}{6} \\ 0.8 & -1 \end{pmatrix} \leftarrow \mathbb{X}_1$$

$$\Phi(\mathbb{X}_2) = 52.57$$

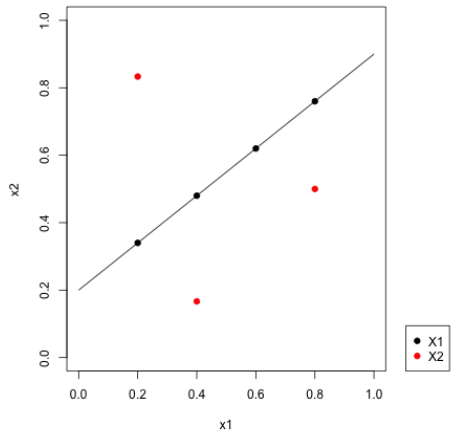
The building algorithm consists in permutating rows inside each column until finding the optimal combination for the criterion Φ. The -1 stays at the bottom of the second column. $\mathbb{X}_2$ is then deduced by keeping only the first 3 rows. The permutation is done on all rows in the first column and on the first three rows in the second column. One example of possible permutations is shown in figure below. Inside the first column (resp. the second column), the permutation $\begin{pmatrix} 4 & 2 & 1 & 3 \end{pmatrix}$ (resp. $\begin{pmatrix} 2 & 3 & 1 \end{pmatrix}$) is carried out.



$$\mathbb{X}_2 \rightarrow \begin{pmatrix} 0.8 & \frac{3}{6} \\ 0.4 & \frac{5}{6} \\ 0.2 & \frac{1}{6} \\ 0.6 & -1 \end{pmatrix}$$

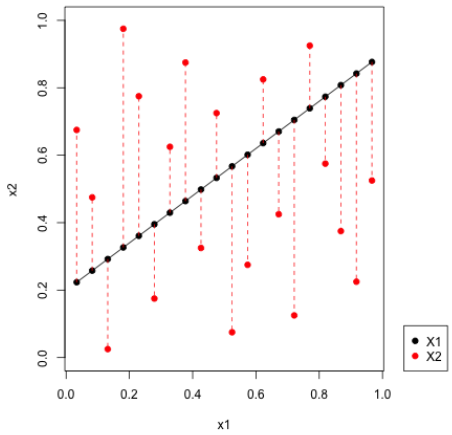
$$\Phi(\mathbb{X}_2) = 7.69$$

The optimal permutations for the criterion Φ are shown in figure below. They are found using algorithm 1 in appendix 9.4.2.

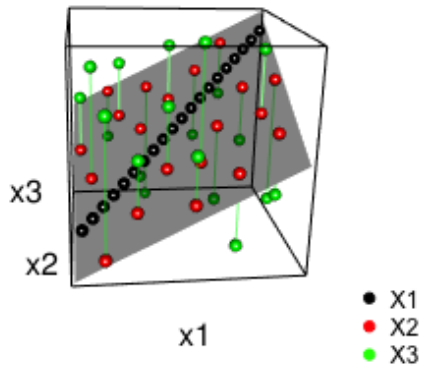


$$\mathbb{X}_2 \rightarrow \begin{pmatrix} 0.2 & \frac{5}{6} \\ 0.8 & \frac{3}{6} \\ 0.4 & \frac{1}{6} \\ 0.6 & -1 \end{pmatrix}$$

$$\Phi(\mathbb{X}_2) = 6.21$$



(a) $\mathbb{X}_2$ nested in $\mathbb{X}_1$



(b) $\mathbb{X}_3$ nested in $\mathbb{X}_2$

Figure 5.2: Nested designs: $\mathbb{X}_1$ in black, $\mathbb{X}_2$ in red, and $\mathbb{X}_3$ in green. Panel 5.2a is a 2D representation of $\mathbb{X}_1$ and $\mathbb{X}_2$. The dotted red lines show that $\mathbb{X}_2$ is nested in $\mathbb{X}_1$ (shares same values of x_1). Panel 5.2b is a 3D representation of $\mathbb{X}_1$, $\mathbb{X}_2$, and $\mathbb{X}_3$. The green lines show that $\mathbb{X}_3$ is nested in $\mathbb{X}_2$ (shares same values of (x_1, x_2)).

The example in 4D is shown in appendix 9.1.3.

5.1.4 Other illustrations

Two examples in dimension 3 are considered, one with three steps and one with two steps.

Three-step example In this first example, the three steps are defined by:

- Step 1: x_1 is free. (x_2, x_3) is fixed equal to $(0.7x_1 + 0.2, \frac{x_1+0.7x_1+0.2}{2})$.
- Step 2: x_1 and x_2 are free. x_3 is fixed to $\frac{x_1+x_2}{2}$.
- Step 3: all variables are free.

Three nested DoE's $\mathbb{X}_1$, $\mathbb{X}_2$, $\mathbb{X}_3$ of size 20 are generated. $\mathbb{X}_1$ is a grid in $[0, 1]$. $\mathbb{X}_2$ and $\mathbb{X}_3$ are built following algorithm 1. They are shown in figure 5.2. The design $\mathbb{X}_2$ (resp. $\mathbb{X}_3$) verifies the nesting property $\mathbb{X}_2 \sqsubset \mathbb{X}_1$ (resp. $\mathbb{X}_3 \sqsubset \mathbb{X}_2$), is well spread in $[0, 1]^2$ (resp. in $[0, 1]^3$), and is far from the subspace $(x_2, x_3) = (0.7x_1 + 0.2, \frac{x_1+0.7x_1+0.2}{2})$ represented by the black line on panel 5.2a (resp. the subspace $x_3 = \frac{x_1+x_2}{2}$ represented by the black plane in panel 5.2b).

Two-step example In this second example, the two steps are defined by:

- Step 1: x_1 is free. (x_2, x_3) is fixed equal to $(0.7x_1 + 0.2, -0.3x_1 + 0.5)$.
- Step 2: all variables are free.

Two nested designs $\mathbb{X}_1$ and $\mathbb{X}_2$ respectively of size 15 and 10 are generated, $\mathbb{X}_1$ composed of equispaced points in $[0, 1]$, and $\mathbb{X}_2$ built following algorithm 1. They are shown in figure 5.3.

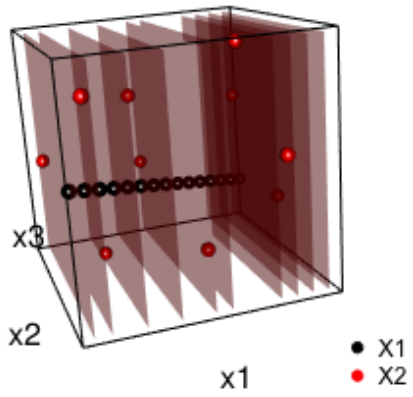


Figure 5.3: 3D representation of the nested DoE's: $\mathbb{X}_1$ in black and $\mathbb{X}_2$ in red. The red plans show that $\mathbb{X}_2$ is nested in $\mathbb{X}_1$ as it shares the same values of x_1.

5.2 Non-nested designs

This section deals with the building method of $\mathbb{X}_n$ ($n \geqslant 2$) in a non-nested training designs context. In this case, the designs $\widetilde{\mathbb{X}}_n$ ($n \geqslant 2$) play the role that was embodied by $(\mathbb{X}_n)_{n=1}^N$ in the previous section as they are nested. $\widetilde{\mathbb{X}}_n$ is defined as the concatenations of the parts of $\mathbb{X}_n$, ..., $\mathbb{X}_N$ corresponding to the variables $x_{I_1 \cup \cdots \cup I_n}$ (see definition 8 in chapter 4). In order to invert the matrices $(cov(Z_n(\widetilde{\mathbb{X}}_n), Z_n(\widetilde{\mathbb{X}}_n)))_{n=1}^N$, the designs $\widetilde{\mathbb{X}}_n$ must have points sufficiently appart from each other (for $n \geqslant 1$) and far from the nullity subspaces (for $n \geqslant 2$). From this wish, optimization problems are defined for the different submatrices of the designs and solved by a simulated annealing algorithm.

5.2.1 Iterative construction procedure

The design $\widetilde{\mathbb{X}}_n$ ($n \in [\![1, N]\!]$) must verify the same constraints than $\mathbb{X}_n$ in section 5.1:

- It must be space-filling in $[0,1]^{d_1+\cdots+d_n}$ and have distinct points sufficiently far from each other.
- It must have points far from the subspace on which Z_n is null (for $n \geqslant 2$).

These two properties are conditions to the invertibility of the covariance matrices.

The design $\mathbb{X}_n$ ($n \in [\![2, N]\!]$) is involved in $\widetilde{\mathbb{X}}_1, \cdots, \widetilde{\mathbb{X}}_n$. In particular, at step n, the following samples $\mathbb{X}_{n+1}, \cdots, \mathbb{X}_N$ do not exist yet and are not taken into account in the samples $\widetilde{\mathbb{X}}_1$, $\cdots$, $\widetilde{\mathbb{X}}_n$. The parts of the samples $\widetilde{\mathbb{X}}_1, \cdots, \widetilde{\mathbb{X}}_n$ taken into account in the building of $\mathbb{X}_n$ (that are formed only using $\mathbb{X}_1, \cdots, \mathbb{X}_n$) are called $\widetilde{\mathbb{X}}_1^n, \cdots, \widetilde{\mathbb{X}}_n^n$. To ensure the properties that must be verified by these samples, the chosen approach is to build $\mathbb{X}_n$ by group of variables: first build the columns corresponding to x_{I_1} (called $\mathbb{X}_{n,I_1}$), then build the columns corresponding to x_{I_2} (called $\mathbb{X}_{n,I_2}$), $\cdots$, then to x_{I_n} (called $\mathbb{X}_{n,I_n}$). Each part of $\mathbb{X}_n$ must solve a particular optimization problem.

- $\mathbb{X}_{n,I_1}$ must solve the following optimization problem

$$\min_{\mathbb{X}_{n,I_1}\subset[0,1]^{d_1}} \Phi_{n,1}(\mathbb{X}_{n,I_1}),$$

with $\Phi_{n,1}$ equal to:

$$\Phi_{n,1}(\mathbb{X}_{n,I_1}) = \left(\sum_{\substack{x,t\in\overset{\sim}{\mathbb{X}}{}^n_1 \\ x\neq t}} \left(\frac{1}{\|x-t\|^2_{[0,1]^{d_1+\cdots+d_n}}} \right)^{50} \right)^{\frac{1}{50}}$$

$\Phi_{n,1}$ is the generalized maximin applied to the sample $\overset{\sim}{\mathbb{X}}{}^n_1$.

- $\mathbb{X}_{n,I_k}$ ($k \in [\![2,n]\!]$) must solve the following optimization problem:

$$\begin{cases} \min\limits_{\mathbb{X}_{n,I_k}\subset[0,1]^{d_1+\cdots+d_k}} \Phi_{n,k}(\mathbb{X}_{n,I_k}), \\ \\ \text{under the constraint: } \mathbb{X}_{n,I_1\cup\cdots\cup I_k} \sqsubset \mathbb{X}_{n,I_1\cup\cdots\cup I_{k-1}}, \end{cases}$$

with $\Phi_{n,k}$ equal to:

$$\Phi_{n,k}(\mathbb{X}_{n,I_k}) = \left(\sum_{\substack{x,t\in\overset{\sim}{\mathbb{X}}{}^n_k \\ x\neq t}} \left(\frac{1}{\|x-t\|^2_{[0,1]^{d_1+\cdots+d_n}}} \right)^{50} \right)^{\frac{1}{50}} + \left(\sum_{x\in\mathbb{X}_{n,I_1\cup\cdots\cup I_k}} \left(\frac{1}{\|x_{I_k}-\grave{x}_{I_k}\|^2_{[0,1]^{d_k}}} \right)^{50} \right)^{\frac{1}{50}}.$$

The first part is the generalized maximin applied to $\overset{\sim}{\mathbb{X}}{}^n_k$, the second part ensures that $\mathbb{X}_{n,I_1\cup\cdots\cup I_k}$ is far from the nullity subspace $\{(x_{I_1\cup\cdots\cup I_{k-1}}, \grave{x}_{I_k}), x_{I_1\cup\cdots\cup I_{k-1}} \in [0,1]^{d_1+\cdots+d_{k-1}}\}$ where Z_k is null almost surely.

5.2.2 Numerical implementation

From $\mathbb{X}_1, \cdots, \mathbb{X}_{n-1}$, $\mathbb{X}_n$ is built block by block.

- If $I_1 = \{1\}$, the matrix $\mathbb{X}_{n,I_1}$ is a column whose components are adjusted by hand to have a good value of $\Phi_{n,1}$. If I_1 is of length greater than 1, the matrix $\mathbb{X}_{n,I_1}$ is created by interverting rows from an initial design $\mathbb{X}^{(0)}$ by a simulated annealing process, in order to minimize the criterion $\Phi_{n,1}(\mathbb{X}_{n,I_1})$. $\mathbb{X}^{(0)}$ is an LHS design of size n_n independent from $\overset{\sim}{\mathbb{X}}{}^{n-1}_1$, with d_1 columns. The simulated annealing algorithm is described in algorithm 2 in appendix 9.4.2.

- For $k \in [\![2,n]\!]$, $\mathbb{X}_{n,I_1\cup\cdots\cup I_k}$ is created by interverting rows from an initial design $\mathbb{X}^{(0)}$ by a simulated annealing process, in order to minimize the criterion $\Phi_{n,k}(\mathbb{X}_{n,I_k})$. Denoting the sample $\mathbb{X}_{n,I_1\cup\cdots\cup I_{k-1}}$ by:

$$\mathbb{X}_{n,I_1\cup\cdots\cup I_{k-1}} = \begin{pmatrix} x^{(1)}_{I_1\cup\cdots\cup I_{k-1}} \\ \vdots \\ x^{(n_n)}_{I_1\cup\cdots\cup I_{k-1}} \end{pmatrix},$$

the initial design is taken equal to:

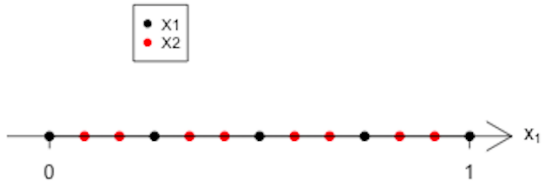
$$\mathbb{X}^{(0)} = \left[\begin{array}{c|ccc} x^{(1)}_{I_1\cup\cdots\cup I_{k-1}} & \frac{1}{2n_n} & \cdots & \frac{1}{2n_n} \\ \vdots & \vdots & & \vdots \\ x^{(i)}_{I_1\cup\cdots\cup I_{k-1}} & \frac{1}{2n_n}+\frac{i-1}{n_n} & \cdots & \frac{1}{2n_n}+\frac{i-1}{n_n} \\ \vdots & \vdots & & \vdots \\ x^{(n_n)}_{I_1\cup\cdots\cup I_{k-1}} & 1-\frac{1}{2n_n} & \cdots & 1-\frac{1}{2n_n} \end{array}\right]$$

The simulated annealing algorithm is again algorithm 1, as in section 5.1, where $\mathbb{X}_{n,I_1\cup\cdots\cup I_k}$ (resp. $\mathbb{X}_{n,I_1\cup\cdots\cup I_{k-1}}$ and $\Phi_{n,k}$) plays the role of $\mathbb{X}_n$ (resp. $\mathbb{X}_{n-1}$ and Φ_n). This time both samples are of the same size, all the rows are kept, and there is no need to intervert rows on the left side.

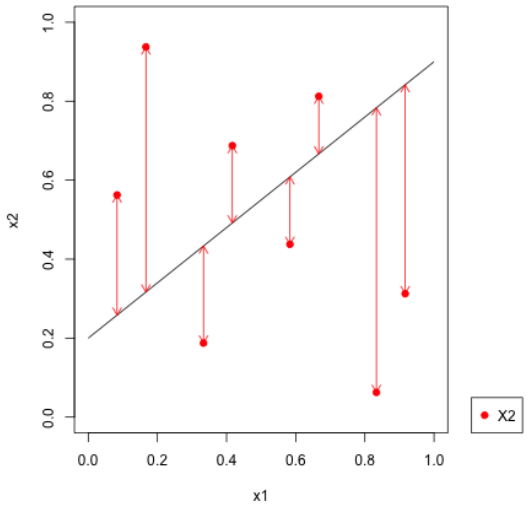
5.2.3 Examples

Example in 2D In the example in 2D from subsection 3.1.2 in chapter 3, $\mathbb{X}_1$ is supposed equal to $\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}$. The goal is to build $\mathbb{X}_2$ such that:

- $\widetilde{\mathbb{X}}_1$ is space filling in $[0,1]$, has distinct points far from each other. This property is illustrated below. All the points of $\widetilde{\mathbb{X}}_1$ are represented in $[0,1]$. A distinction is made between the points coming from $\mathbb{X}_1$ in black, and those coming from $\mathbb{X}_2$ in red.



- $\widetilde{\mathbb{X}}_2 = \mathbb{X}_2$ is space-filling in $[0,1]^2$, has distinct points far from each other and far from the nullity subspace $\mathcal{D} = \{(x_1, 0.7x_1 + 0.2), x_1 \in [0,1]\}$. This design is illustrated below with red points, their distance to the nullity subspace is shown in red arrow.



The chosen approach is to build $\mathbb{X}_2$ one column at a time.

- The first column $\mathbb{X}_{2,1} = \begin{pmatrix} x_1^{(n_1+1)} \\ \vdots \\ x_1^{(n_1+n_2)} \end{pmatrix}$ must solve the following optimization problem

$$\min_{\mathbb{X}_{2,1} \subset [0,1]} \Phi_{2,1}(\mathbb{X}_{2,1}),$$

with $\Phi_{2,1}$ equal to:

$$\Phi_{2,1}(\mathbb{X}_{2,1}) = \left(\sum_{i=1}^{n_1+n_2} \sum_{\substack{j=1 \\ j \neq i}}^{n_1+n_2} \left(\frac{1}{(x_1^{(i)} - x_1^{(j)})^2} \right)^{50} \right)^{\frac{1}{50}}$$

- When the first column is created, the problem is reduced to the one of the previous subsection, of generating a design $\mathbb{X}_2$ nested in $\mathbb{X}_{2,1}$. The second column $\mathbb{X}_{2,2} = \begin{pmatrix} x_2^{(n_1+1)} \\ \vdots \\ x_2^{(n_1+n_2)} \end{pmatrix}$ must solve the following optimization problem:

$$\begin{cases} \min_{\mathbb{X}_{2,2} \subset [0,1]} \Phi_{2,2}(\mathbb{X}_{2,2}), \\ \\ \text{under the constraint: } \mathbb{X}_2 \sqsubset \mathbb{X}_{2,1}, \end{cases}$$

with $\Phi_{2,2}$ equal to:

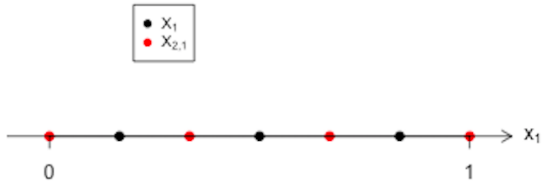
$$\begin{aligned} \Phi_{2,2}(\mathbb{X}_{2,2}) \quad = \quad & \left(\sum_{i=n_1+1}^{n_1+n_2} \sum_{\substack{j=n_1+1 \\ j \neq i}}^{n_1+n_2} \left(\frac{1}{(x_1^{(i)} - x_1^{(j)})^2 + (x_2^{(i)} - x_2^{(j)})^2} \right)^{50} \right)^{\frac{1}{50}} \\ + \quad & \left(\sum_{i=n_1+1}^{n_1+n_2} \left(\frac{1}{(x_2^{(i)} - 0.7 x_1^{(i)} - 0.2)^2} \right)^{50} \right)^{\frac{1}{50}}. \end{aligned}$$

To illustrate the building method, it is assumed that a design $\mathbb{X}_1 = \begin{pmatrix} \frac{1}{6} \\ \frac{3}{6} \\ \frac{5}{6} \end{pmatrix}$ has been generated at step 1. The variable x_2 is implicitely fixed to $0.7x_1 + 0.2$. The goal is now to build $\mathbb{X}_2$ of size 4 with x_2 that can take its values freely in $[0, 1]$.

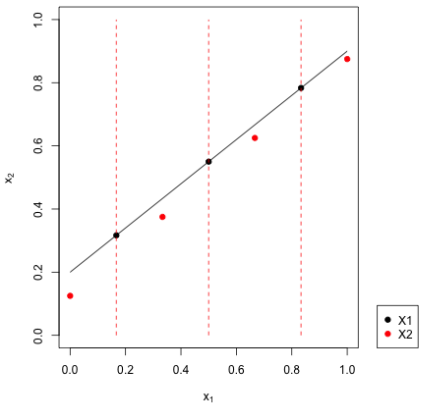
- The components of the first column $\mathbb{X}_{2,1}$ are adjusted by hand so that $\widetilde{\mathbb{X}}_1$ has all the required properties. For example, $\mathbb{X}_{2,1}$ is chosen equal to:

$$\mathbb{X}_{2,1} = \begin{pmatrix} \frac{0}{6} \\ \frac{2}{6} \\ \frac{4}{6} \\ \frac{6}{6} \end{pmatrix}.$$

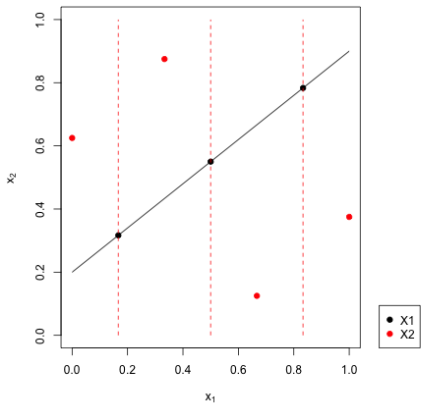
  The corresponding $\widetilde{\mathbb{X}}_1$ is represented below.



- Then the second column is built so that $\mathbb{X}_2$ is nested in $\mathbb{X}_{2,1}$. It is initialized to $\mathbb{X}_2 = \begin{pmatrix} \frac{0}{6} & \frac{1}{8} \\ \frac{2}{6} & \frac{3}{8} \\ \frac{4}{6} & \frac{5}{8} \\ \frac{6}{6} & \frac{7}{8} \end{pmatrix}$. The optimal design for the criterion Φ_2 is obtained for the permutation $\begin{pmatrix} 3 & 4 & 1 & 2 \end{pmatrix}$ in the second column, giving the design: $\mathbb{X}_2 = \begin{pmatrix} \frac{0}{6} & \frac{5}{8} \\ \frac{2}{6} & \frac{7}{8} \\ \frac{4}{6} & \frac{1}{8} \\ \frac{6}{6} & \frac{3}{8} \end{pmatrix}$. It is found using algorithm 1 in appendix 9.4.2. The two designs are illustrated below.



Initial design: $\Phi_2(\mathbb{X}_2) = 42.45$

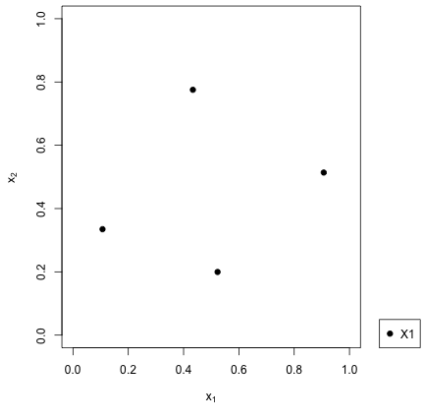


Optimal design: $\Phi_2(\mathbb{X}_2) = 5.05$

The example in 4D is shown in appendix 9.1.3.

Example in 3D In the examples in 2D (see paragraph above) and 4D (see appendix 9.1.3), the case I_1 of length greater than 1 is not tackled. So this case is illustrated in the following example in 3D. In this example, the first two variables are released at step 1 ($I_1 = \{1, 2\}$), x_3 being fixed to 0.6 and then released at step 2 ($I_2 = \{3\}$). Designs $\mathbb{X}_1$ of size 4 and $\mathbb{X}_2$ of size 5 must be created respectively at step 1 and 2.

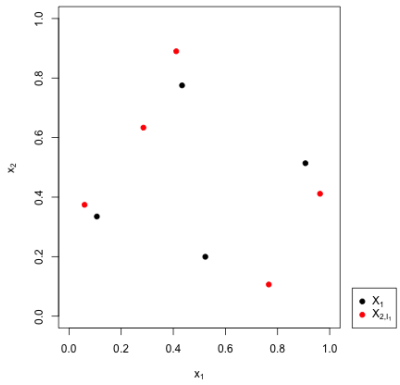
- At step 1, the DoE $\mathbb{X}_1$ is taken as an LHS optimized for the generalized maximin criterion:



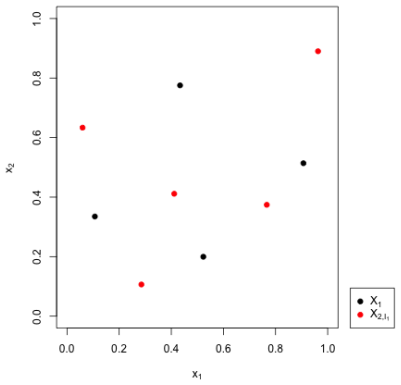
$$\mathbb{X}_1 = \begin{pmatrix} 0.91 & 0.51 \\ 0.11 & 0.33 \\ 0.52 & 0.20 \\ 0.43 & 0.78 \end{pmatrix}$$

- At step 2, the goal is to create the design $\mathbb{X}_2$ of size 5.
  - The submatrix $\mathbb{X}_{2,I_1}$ composed of the first two columns of $\mathbb{X}_2$ is build in first. It is initialized to an LHS: $\mathbb{X}_{2,I_1} = \begin{pmatrix} 0.96 & 0.41 \\ 0.06 & 0.37 \\ 0.29 & 0.63 \\ 0.77 & 0.11 \\ 0.41 & 0.89 \end{pmatrix}$. The submatrix $\mathbb{X}_{2,I_1}$ must be optimal for the criterion $\Phi_{2,1}$ which is the generalized maximin criterion applied

to $\overset{\sim 2}{\mathbb{X}}_2 = \begin{pmatrix} 0.91 & 0.51 \\ 0.11 & 0.33 \\ 0.52 & 0.20 \\ 0.43 & 0.78 \\ 0.96 & 0.41 \\ 0.06 & 0.37 \\ 0.29 & 0.63 \\ 0.77 & 0.11 \\ 0.41 & 0.89 \end{pmatrix}$ (the concatenation of $\mathbb{X}_1$ and $\mathbb{X}_{2,I_1}$). The optimal design is obtained for the permutation $\begin{pmatrix} 2 & 1 & 4 & 3 & 5 \end{pmatrix}$ in the first column of $\mathbb{X}_{2,I_1}$ and for the permutation $\begin{pmatrix} 3 & 5 & 2 & 4 & 1 \end{pmatrix}$ in its second column: $\mathbb{X}_{2,I_1} = \begin{pmatrix} 0.06 & 0.63 \\ 0.96 & 0.89 \\ 0.77 & 0.37 \\ 0.29 & 0.11 \\ 0.41 & 0.41 \end{pmatrix}$. It is found using algorithm 2 in appendix 9.4.2. The two designs are shown below.



Initial design:
$\Phi_{2,1}(\mathbb{X}_{2,I_1}) = 16.26$

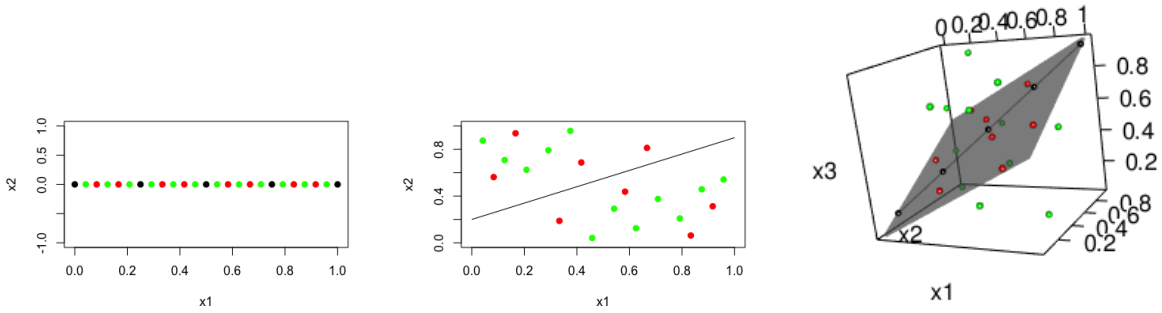


Optimal design:
$\Phi_{2,1}(\mathbb{X}_{2,I_1}) = 5.05$

– $\mathbb{X}_2$ is then built as a design in dimension 3 nested in $\mathbb{X}_{2,I_1}$. The goal is to built its third column. The design is initialized to

$$\mathbb{X}_2 = \begin{pmatrix} 0.06 & 0.63 & \frac{1}{10} \\ 0.96 & 0.89 & \frac{3}{10} \\ 0.77 & 0.37 & \frac{5}{10} \\ 0.29 & 0.11 & \frac{7}{10} \\ 0.41 & 0.41 & \frac{9}{10} \end{pmatrix}, \quad \Phi_{2,2}(\mathbb{X}_2) = 12.73.$$

The optimal design for the criterion $\Phi_{2,2}$ is obtained for the permutation $\begin{pmatrix} 3 & 2 & 4 & 5 & 1 \end{pmatrix}$ in the last column. It is found using algorithm 1 in appendix 9.4.2.



(a) Projections of $\mathbb{X}_1$, $\mathbb{X}_2$, $\mathbb{X}_3$ on x_1

(b) Projections of $\mathbb{X}_2$, $\mathbb{X}_3$ on (x_1, x_2)

(c) $\mathbb{X}_1$, $\mathbb{X}_2$, $\mathbb{X}_3$ samples in $3D$

Figure 5.4: $\mathbb{X}_1$ is represented by the black points. It is the sample from step 1 when $x_2 = 0.7x_1 + 0.2$ and $x_3 = \frac{x_1+0.7x_1+0.2}{2}$ (black line). $\mathbb{X}_2$ is represented by the red points. It is the sample from step 2 when x_2 is released and $x_3 = \frac{x_1+x_2}{2}$. $\mathbb{X}_3$ is represented by the green points. It is the sample from step 3 when all inputs are released. Its projections on the plane together with the $\mathbb{X}_2$ points form a space filling design of $[0,1]^2$ (see panel 5.4b). Its projections on the line together with $\mathbb{X}_1$ and $\mathbb{X}_2$ points form a space filling design of $[0,1]$ (see panel 5.5a). $\mathbb{X}_1$, $\mathbb{X}_2$, and $\mathbb{X}_3$ are represented in 3D in panel 5.4c

$$\mathbb{X}_2 = \begin{pmatrix} 0.06 & 0.63 & \frac{5}{10} \\ 0.96 & 0.89 & \frac{3}{10} \\ 0.77 & 0.37 & \frac{7}{10} \\ 0.29 & 0.11 & \frac{9}{10} \\ 0.41 & 0.41 & \frac{1}{10} \end{pmatrix}, \quad \Phi_{2,2}(\mathbb{X}_2) = 11.89.$$

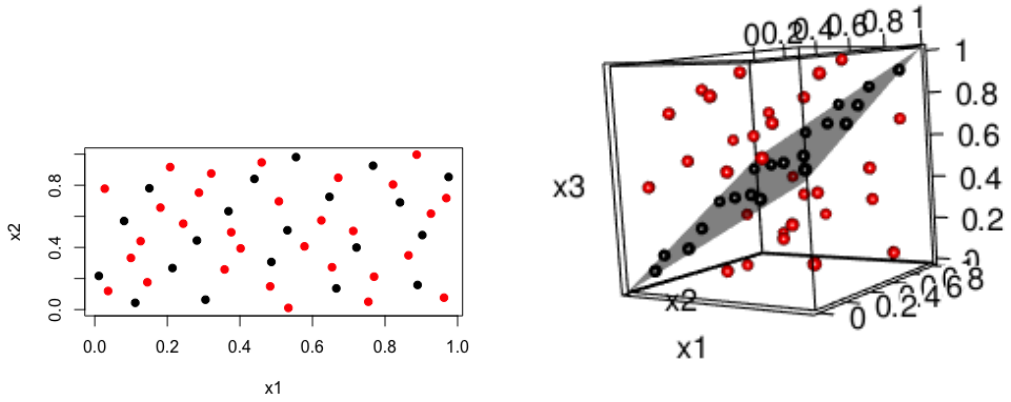
5.2.4 Other illustrations

Two examples in dimension 3 are considered, one with three steps and one with two steps.

Three-step example It is the same three-step example as in section 5.1 but this time $\mathbb{X}_1$ is taken of size 5, $\mathbb{X}_2$ of size 8, and $\mathbb{X}_3$ of size 12, which is not enabled in the previous case as the size of $\mathbb{X}_n$ cannot increase with n. See figure 5.4. Here the algorithm 2 is not used as the x_1 values of the different samples are just taken from an equispaced discretization of $[0,1]$.

Two-step example In this example there are three inputs (x_1, x_2, x_3).

- At step 1, x_1 and x_2 are free, x_3 is fixed equal to $\frac{x_1+x_2}{2}$.
- At step 2, all inputs are free.



(a) Projections of $\mathbb{X}_1$ and $\mathbb{X}_2$ on (x_1, x_2) (b) $\mathbb{X}_1$ and $\mathbb{X}_2$ samples in $3D$

Figure 5.5: $\mathbb{X}_1$ is represented by the black points. It is the sample from step 1 when $x_3 = \frac{x_1+x_2}{2}$ (black plane). $\mathbb{X}_2$ is represented by the red points. It is the sample from step 2 when x_3 is released. Its projections on the plane together with the $\mathbb{X}_1$ points form a space filling design of $[0,1]^2$. $\mathbb{X}_1$ and $\mathbb{X}_2$ are represented in the 3D input space on panel 5.5b. Their $2D$ projections on the plane are shown on panel 5.5a.

$\mathbb{X}_1$ is of size 30 and $\mathbb{X}_2$ of size 20. See figure 5.5. This time, algorithm 2 has been used to attribute the values of (x_1, x_2) to $\mathbb{X}_2$ such that its projection, together with $\mathbb{X}_1$, forms a space filling design with remote points in $[0,1]^2$.

5.3 Conclusion

In this chapter, a building method is suggested to generate nested samples for the seqGPR metamodel. From the wish to have designs with points sufficiently appart from each other and far from the nullity subspaces, an optimization problem is defined and solved by a simulated annealing algorithm. Two illustrations are then shown of the designs resulting from this algorithm: one example with three steps respectively in 1D, 2D and 3D, and one example with two steps respectively in 1D and 3D. Nested designs have the advantage of making the parameter estimation of the seqGPR metamodel easier as the loss function can be decoupled, however they limit the exploration of the input space as their building is very constrained, and their size cannot increase with the number of the step which is problematic with the increasing dimension of the subspace considered.

A building method is also suggested to generate non-nested samples for the seqGPR metamodel. From the wish to have designs with points sufficiently appart from each other and far from the nullity subspaces, optimization problems are defined for the different submatrices of the designs and solved by a simulated annealing algorithm. Two illustrations are then shown of the designs resulting from this algorithm: one example with three steps respectively in 1D, 2D and 3D, and one example with two steps respectively in 2D and 3D. Non-nested designs have the advantage of being more flexible: no constraint on the size and more possibility to explore the input space. However the parameter estimation is more complicated as the loss

function is not decoupled. This problem is solved in subsection 4.1.2 in chapter 4 with the use of the EM algorithm.

Chapter 6

Additional contributions

Previous chapters define the seqGPR metamodel (caracterized by the equation (3.1) in chapter 3), build its underlying processes, suggest estimation methods for its parameters, and propose methodologies to generate its DoE's. This chapter introduces new works on the seqGPR metamodel to try to answer some unsolved issues. Section 6.1 tackles the case where multiple designs are generated at a given step for different values of the fixed variables. This issue implies a new model than (3.1) as this previous model concerns only the case with one design per step, associated with one value of the fixed variables. The goal of this section is to build this new model and see if it performs as well as the classic kriging model on the test cases of chapter 4. Section 6.2 deals with the enrichment of the training samples of the seqGPR metamodel. Here, the original model (3.1) is considered. After having fitted the seqGPR metamodel on the samples, the goal is to add relevant points to the samples to improve its prediction. In particular, a criterion to use in the search for enrichment points is proposed and the enrichment procedure is evaluated on the test cases of chapter 4.

6.1 Conditioning on multiple subspaces

This section describes the case where, at each step, several subspaces corresponding to different values of the fixed inputs are considered (instead of one as that was the case until now). The adapted seqGPR model is detailed and new candidates for the $(Z_n)_{n\geqslant 2}$, which verify a new nullity property, are suggested. The new metamodel is then compared to classic kriging on analytic test cases.

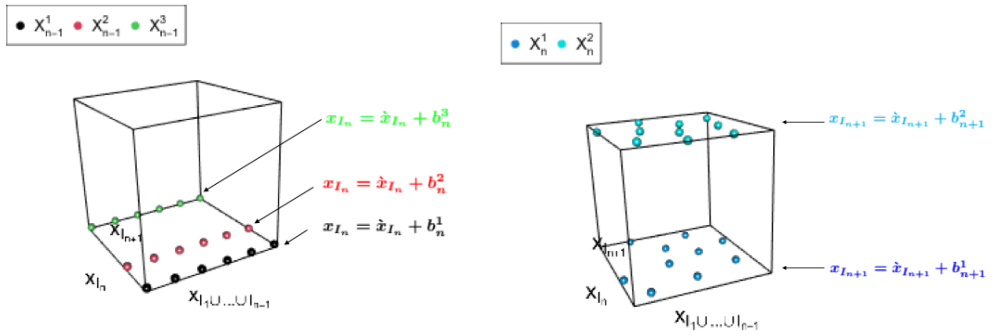
6.1.1 Model

Let $f(x_{I_1}, \cdots, x_{I_N})$ be the output to approximate by a metamodel. Let

$$\mathbb{B}_2 = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{b_2^2} \\ \vdots \\ \boldsymbol{b_2^{K_2}} \end{pmatrix} \subset [0,1]^{d_2}, \quad \mathbb{B}_3 = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{b_3^2} \\ \vdots \\ \boldsymbol{b_3^{K_3}} \end{pmatrix} \subset [0,1]^{d_3}, \quad \cdots, \quad \mathbb{B}_N = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{b_N^2} \\ \vdots \\ \boldsymbol{b_N^{K_N}} \end{pmatrix} \subset [0,1]^{d_N},$$

be matrices such that $\mathbb{B}_n$ contains the constant values which constitute the difference between the subspaces at step $n-1$. The DoE's and subspaces considered are illustrated in figure 6.1.

The problem needs a new model which is different from (3.1) in chapter 3. At step $n \in [\![1, N-1]\!]$, several subspaces are considered, corresponding to different values of $x_{I_{n+1}}$: $\grave{x}_{I_{n+1}}$,



(a) Step $n-1$ (b) Step n

Figure 6.1: Illustration of the designs in the subspaces at step $n-1$ in panel 6.1a, and at step n in panel 6.1b. Here, there are three subspaces at step $n-1$, so $K_n = 3$, and there are two subspaces at step n, so $K_{n+1} = 2$.

$\grave{x}_{I_{n+1}} + b^2_{n+1}$, ..., $\grave{x}_{I_{n+1}} + b^{K_{n+1}}_{n+1}$. As a consequence, the process Y_n modeling the output at step n must also be function of $x_{I_{n+1}}$. The problem is modeled by the following statistical model:

$$\left\{ \begin{array}{lcl} Y_1(x_{I_1}, x_{I_2}) & = & m + Z_1(x_{I_1}, x_{I_2}), \\ Y_n(x_{I_1 \cup \cdots \cup I_n}, x_{I_{n+1}}) & = & Y_{n-1}(x_{I_1 \cup \cdots \cup I_n}) + Z_n(x_{I_1 \cup \cdots \cup I_n}, x_{I_{n+1}}), \\ Y_N(x_{I_1}, \cdots, x_{I_N}) & = & Y_{N-1}(x_{I_1}, \cdots, x_{I_N}) + Z_N(x_{I_1}, \cdots, x_{I_N}), . \end{array} \right. \tag{6.1}$$

where:

- The processes $(Z_n)_{1 \leqslant n \leqslant N}$ are centered independent Gaussian processes of covariance kernel $(\sigma_n^2 \rho_n)_{n=1}^N$
- They must verify the following property:

$$\left\{ \begin{array}{lcll} Z_n(x_{I_1 \cup \cdots I_{n-1}}, \grave{x}_{I_n} + \mathbb{B}_n, \grave{x}_{I_{n+1}}) & = & \mathbf{0}, & \forall n \in [\![2, N-1]\!], \forall x_{I_1 \cup \cdots I_{n-1}} \in [0,1]^{d_1 + \cdots + d_{n-1}}, \\ Z_N(x_{I_1 \cup \cdots \cup I_{N-1}}, \grave{x}_{I_N} + \mathbb{B}_N) & = & \mathbf{0}, & \forall x_{I_1 \cup \cdots \cup I_{N-1}} \in [0,1]^{d_1 + \cdots + d_{N-1}}. \end{array} \right. \tag{6.2}$$

Estimation and prediction of this metamodel is not different from the methodology described in chapter 4. However, the nullity property (6.2) is more complicated than (3.3), as it is on multiple subspaces and not just one. The definition of a correction process Z_n ($n \geqslant 2$) verifying it is the object of the next subsection.

6.1.2 Candidate for the correction processes

The goal of this subsection is to build a process $Z : [0,1]^{d_J + d_I} \times \Omega \to \mathbb{R}$ such that:

$$Z(x_J, g(x_J) + \mathcal{B}) = \mathbf{0} \quad \forall x_J \in [0,1]^{d_J}, \tag{6.3}$$

with $g : [0,1]^{d_J} \to [0,1]^{d_I}$ a deterministic function, and $\mathcal{B} = \{b_1, \cdots, b_K\}$ a finite subset of $[0,1]^{d_I}$. $Z(x_J, g(x_J) + \mathcal{B})$ is the vector of size K whose component i ($i \in [\![1, K]\!]$) is equal to $Z(x_J, g(x_J) + b_i)$.

The proposed candidate is derived from the P process which is the easiest to generalize to this case (see subsection 3.2.3 in chapter 3). It is built based on a latent process $\widetilde{Z} \sim$

$\mathcal{GP}\left(0, \sigma^2 r((x_J, x_I), (t_J, t_I))\right)$ of covariance kernel $\sigma^2 r$. It is defined as:

$$Z^P(x_J, x_I) = \widetilde{Z}(x_J, x_I) - \mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid Z(t_J, g(t_J) + b),\ \forall t_J \in [0,1]^{d_J},\ \forall b \in \mathcal{B}\right],$$

where $\mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid Z(t_J, g(t_J) + b)\ \forall t_J \in [0,1]^{d_J},\ \forall b \in \mathcal{B}\right]$ is the projection of $\widetilde{Z}(x_J, x_I)$ on the sub-Gaussian space engendered by the family $\{\widetilde{Z}(t_J, g(t_J) + b),\ t_J \in [0,1]^{d_J},\ b \in \mathcal{B}\}$. Its general expression is given in subsection 2.2.2 in chapter 2 and is recalled below:

$$\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathcal{D})\right] = \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}(x_s) \mathrm{d}\nu(s), \tag{6.4}$$

with $\mathcal{D} = \{x_s, s \in \mathcal{S}\}$, ν a measure on $\mathcal{S}$, and

$$\phi_n(x) = \tfrac{1}{\lambda_n} \textstyle\int_{\mathcal{S}} \sigma^2 r(x, x_s) \widetilde{\phi}_n(s) \mathrm{d}\nu(s).$$

$(\lambda_n, \widetilde{\phi}_n)_{n \geqslant 1}$ are solutions of the eigen problem:

$$\int_{\mathcal{S}} \sigma^2 r\left(x_s, x_u\right) \widetilde{\phi}_n(u) \mathrm{d}\nu(u) = \lambda_n \widetilde{\phi}_n(s),\ \forall s \in \mathcal{S},$$

such that

$$\int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{\phi}_m(s) \mathrm{d}\nu(s) = \delta_{nm}\ \forall n, m \geqslant 1,$$

where δ_{nm} is the Kronecker symbol. It is applied to the case:

$$\begin{cases} \mathcal{D} = \{(s_J, g(s_J) + s_I), s = (s_J, s_I) \in \mathcal{S}\}, \\ \mathcal{S} = [0,1]^{d_J} \times \mathcal{B} = \{(s_J, s_I), s_J \in [0,1]^{d_J}, s_I \in \mathcal{B}\}, \\ \nu = \lambda \otimes \left(\sum_{i=1}^{K} \delta_{b_i}\right), \\ x_s = (s_J, g(s_J) + s_I), \end{cases}$$

with λ the Lebesgues measure on $[0,1]^{d_J}$ and δ_{b_i} the Dirac measure at b_i.

Proposition 13 *Let r_J and r_I be two stationary correlation kernels. If r is of the form:*

$$r((x_J, x_I), (t_J, t_I)) = r_J(x_J, t_J) r_I(x_I - g(x_J), t_I - g(t_J)),$$

then the process Z^P is a centered Gaussian process equal to:

$$Z^P(x_J, x_I) = \widetilde{Z}(x_J, x_I) - r_I(x_I - g(x_J), \mathcal{B}) r_I(\mathcal{B}, \mathcal{B})^{-1} \widetilde{Z}(x_J, g(x_J) + \mathcal{B}),\ \forall (x_J, x_I) \in [0,1]^{d_J + d_I}.$$

whose covariance kernel is $\sigma^2 \rho^P$ with:

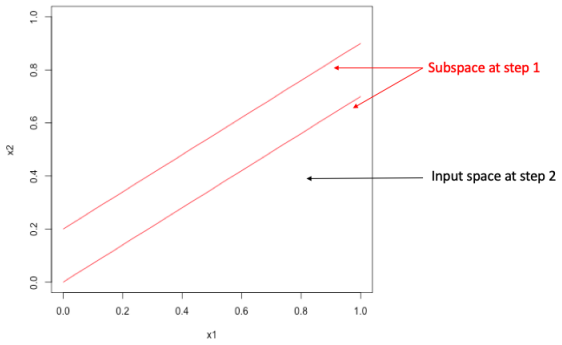
$$\begin{aligned} \rho^P((x_J, x_I), (t_J, t_I)) &= r_J(x_J, t_J) \left[r_I(x_I - g(x_J), t_I - g(t_J)) \right. \\ &- \left. r_I(x_I - g(x_J), \mathcal{B}) r_I(\mathcal{B}, \mathcal{B})^{-1} r_I(\mathcal{B}, t_I - g(t_J))\right]. \end{aligned}$$

Proof *See appendix 9.3.3 for a proof of this proposition.* ∎

6.1.3 Example in 2D

The example in 2D from subsection 3.1.1 in chapter 3 is here adapted to the multi-conditioning case. A two-step study is carried out:

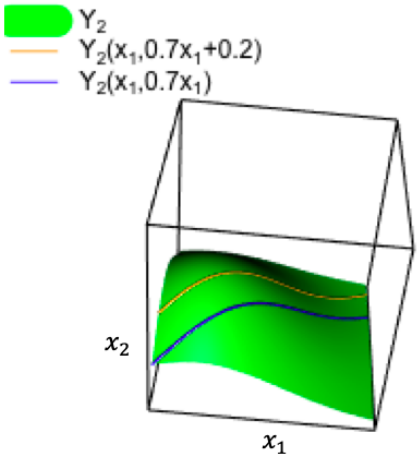
- At step 1, two subspaces are considered: $x_2 = 0.7x_1 + 0.2$ and $x_2 = 0.7x_1$.



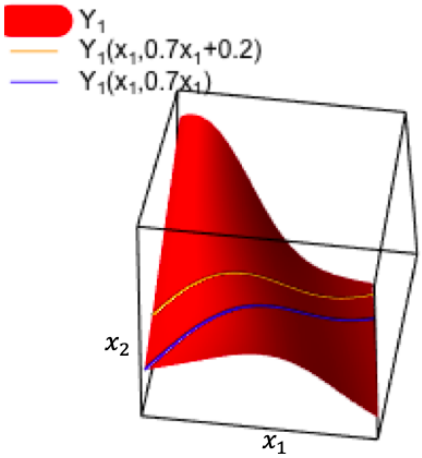
Here, each subspace is indexed by x_1: $\{(x_1, 0.7x_1{+}0.2), x_1 \in [0,1]\}$ and $\{(x_1, 0.7x_1), x_1 \in [0,1]\}$. The restrictions of f on these two subspaces $f(x_1, 0.7x_1 + 0.2)$ and $f(x_1, 0.7x_1)$ need to be approximated. These restrictions are modeled by one Gaussian process $Y_1(x_1, x_2) = m + Z_1(x_1, x_2)$ with m a scalar equal to the mean of Y_1, and Z_1 a centered Gaussian process of covariance kernel $k_1((x_1, x_2), (t_1, t_2))$. The difference with the previous model is that Y_1 is also a function of x_2. This process is such that $f(x_1, 0.7x_1 + 0.2)$ (resp. $f(x_1, 0.7x_1)$) is the realization of its restriction on the first subspace $Y_1(x_1, 0.7x_1 + 0.2)$ (resp. on the second subspace $Y_1(x_1, 0.7x_1)$). Two DoE's are generated: $\mathbb{X}_1^1 = \begin{pmatrix} x_1^{(1,1)} & 0.7x_1^{(1,1)} + 0.2 \\ \vdots & \vdots \\ x_1^{(1,n_1)} & 0.7x_1^{(1,n_1^1)} + 0.2 \end{pmatrix}$ belonging to the first subspace and $\mathbb{X}_1^2 = \begin{pmatrix} x_1^{(1,n_1^1+1)} & 0.7x_1^{(1,n_1^1+1)} \\ \vdots & \vdots \\ x_1^{(1,n_1^1+n_1^2)} & 0.7x_1^{(1,n_1^1+n_1^2)} \end{pmatrix}$ belonging to the second subspace. The values of f at the points of $\mathbb{X}_1^1$ are known, equal to $\boldsymbol{y_1^1} = \begin{pmatrix} f_1(x_1^{(1,1)}, 0.7x_1^{(1,1)} + 0.2) \\ \vdots \\ f_1(x_1^{(1,n_1^1)}, 0.7x_1^{(1,n_1^1)} + 0.2) \end{pmatrix}$. Similarly, the values of f at the points of $\mathbb{X}_1^2$ are known, equal to $\boldsymbol{y_1^2} = \begin{pmatrix} f_1(x_1^{(1,n_1^1+1)}, 0.7x_1^{(1,n_1^1+1)}) \\ \vdots \\ f_1(x_1^{(1,n_1^1+n_1^2)}, 0.7x_1^{(1,n_1^1+n_1^2)}) \end{pmatrix}$.

- At step 2, the entire input space $[0,1]^2$ is considered. The function to approximate is now $f_2(x_1, x_2) = f(x_1, x_2)$. It is modeled by a Gaussian process $Y_2(x_1, x_2)$ linked to $Y_1(x_1, x_2)$ by the formula $Y_2(x_1, x_2) = Y_1(x_1, x_2) + Z_2(x_1, x_2)$. This formula looks like the multifidelity metamodel, but the difference is that Y_1 and Y_2 model the same level of fidelity. Thus, they must coincide on the subspaces $x_2 = 0.7x_1 + 0.2$ and $x_2 = 0.7x_1$. This coincidence, shown in the figures below, implies that $Z_2(x_1, 0.7x_1 + 0.2) = 0$ and $Z_2(x_1, 0.7x_1) = 0$. Figures on the top show Y_2 (on the left) and Y_1 (on the right) and

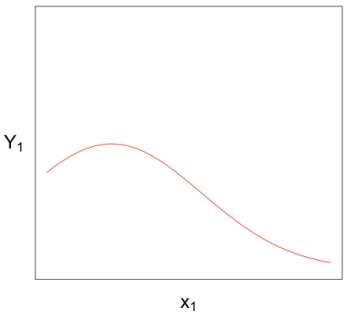
their coincidence on the first subspace in orange and on the second subspace in blue. Their restriction on these subspaces are illustrated on the bottom figures (subspace $x_2 = 0.7x_1 + 0.2$ on the left and $x_2 = 0.7x_1$ on the right).



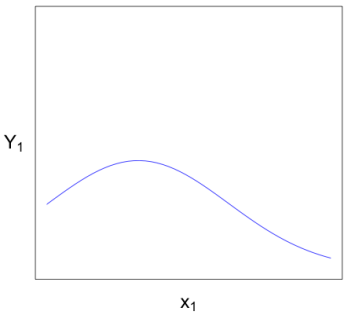
Visualization of Y_2



Y_1 as a function of (x_1, x_2)

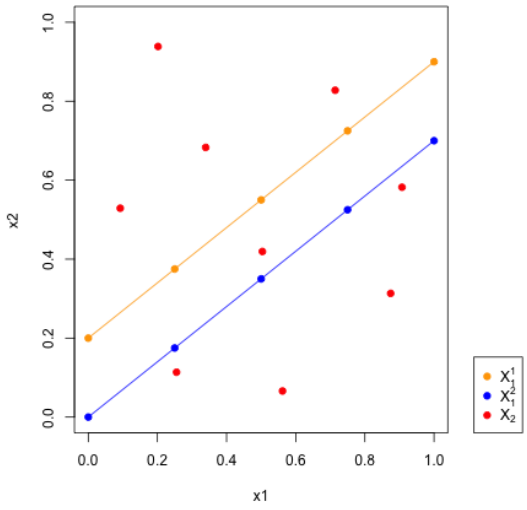


First restriction of Y_1



Second restriction of Y_1

A DoE is generated $\mathbb{X}_2 = \begin{pmatrix} x_1^{(2,1)} & x_2^{(2,1)} \\ \vdots & \vdots \\ x_1^{(2,n_2)} & x_2^{(2,n_2)} \end{pmatrix}$ containing values of (x_1, x_2). The values of f_2 at the points of $\mathbb{X}_2$ are known, equal to $\boldsymbol{y_2} = \begin{pmatrix} f_2(x_1^{(2,1)}, x_2^{(2,1)}) \\ \vdots \\ f_2(x_1^{(2,n_2)}, x_2^{(2,n_2)}) \end{pmatrix}$. An example of the DoE's $\mathbb{X}_1^1$, $\mathbb{X}_1^2$ and $\mathbb{X}_2$ is shown in the figure below:



The samples $\mathbb{X}_1^1$, $\mathbb{X}_1^2$, $\mathbb{X}_2$ belong to the same space $[0,1]^2$. To apply the EM algorithm, a new definition of $\widetilde{\mathbb{X}}_1$ and $\widetilde{\mathbb{X}}_2$ is used, as shown below:

$$\mathbb{X}_1 = \begin{pmatrix} x_1^{(1,1)} & 0.7x_1^{(1,1)} + 0.2 \\ \vdots & \vdots \\ x_1^{(1,n_1^1)} & 0.7x_1^{(1,n_1^1)} + 0.2 \\ x_1^{(1,n_1^1+1)} & 0.7x_1^{(1,n_1^1+1)} \\ \vdots & \vdots \\ x_1^{(1,n_1^1+n_1^2)} & 0.7x_1^{(1,n_1^1+n_1^2)} \end{pmatrix} \quad \widetilde{\mathbb{X}}_1$$

$$\mathbb{X}_2 = \begin{pmatrix} x_1^{(2,1)} & x_2^{(2,1)} \\ \vdots & \vdots \\ x_1^{(2,n_2)} & x_2^{(2,n_2)} \end{pmatrix} \quad \widetilde{\mathbb{X}}_2$$

The goal is to find a candidate for the process $Z_2(x_1, x_2)$, modeling the piece of information added at step 2 by releasing x_2 which was fixed to $0.7x_1 + 0.2$ or $0.7x_1$ at step 1. On the nullity subspaces $\mathcal{D}_1 = \{(x_1, 0.7x_1 + 0.2), x_1 \in [0,1]\}$ and $\mathcal{D}_2 = \{(x_1, 0.7x_1), x_1 \in [0,1]\}$, Z_2 adds no information and must be equal to 0:

$$\begin{cases} Z_2(x_1, 0.7x_1 + 0.2) &= 0, \quad \forall x_1 \in [0,1], \\ Z_2(x_1, 0.7x_1) &= 0, \quad \forall x_1 \in [0,1]. \end{cases}$$

How to find a Gaussian process Z_2 verifying this nullity property and having a computable covariance kernel ? The idea suggested in what follows is to build Z_2 by a transformation of a latent centered Gaussian process $\widetilde{Z}_2$ whose kernel $\sigma_2^2 r_2$ is known and computable. The transformation must impose the nullity property, keep the law of the process Gaussian, and keep the computability of the kernel. The transformation consists in conditioning the latent process to be null on the reunion of the nullity subspaces $\mathcal{D}_1$ and $\mathcal{D}_2$. As this nullity subspace is an

infinite continuous set of points, the usual conditioning seen in proposition 2 cannot be used directly. Instead, a generalization of the conditioning, defined in [Gauthier and Bay, 2012b], is used. The resulting process, derived from the P process, is defined as:

$$Z_2(x_1,x_2) = \widetilde{Z}_2(x_1,x_2) - \mathbb{E}\left[\widetilde{Z}_2(x_1,x_2) \mid \widetilde{Z}_2(t_1,0.7t_1+0.2), \widetilde{Z}_2(t_1,0.7t_1) \forall t_1 \in [0,1]\right],$$

where $\mathbb{E}\left[\widetilde{Z}_2(x_1,x_2) \mid \widetilde{Z}_2(t_1,0.7t_1+0.2), \widetilde{Z}_2(t_1,0.7t_1) \forall t_1 \in [0,1]\right]$ (denoted by $E(x_1,x_2)$ to simplify notations) is itself a generalization of the conditional expectation, defined as the projection of $\widetilde{Z}_2(x_1,x_2)$ in the subGaussian space generated by $\left(\widetilde{Z}_2(t_1,0.7t_1+0.2), \widetilde{Z}_2(t_1,0.7t_1)\right)_{t_1\in[0,1]}$. The formula of the expectation, taken from the proposition 7 and recalled in the next paragraph, is here adapted to the case $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 = \{(s_1, 0.7s_1+s_2), s = (s_1,s_2) \in [0,1]\times\{0,0.2\}\}$, $\mathcal{S} = [0,1]\times\{0,0.2\}$, and ν any measure on $\mathcal{S}$:

$$E(x_1,x_2) = \sum_{n=1}^{+\infty} \phi_n(x_1,x_2) \int_{s\in\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}_2(s_1, 0.7s_1+s_2) \mathrm{d}\nu(s),$$

where:

$$\phi_n(x_1,x_2) = \frac{1}{\lambda_n} \int_{s\in\mathcal{S}} \sigma_2^2 r_2((x_1,x_2),(s_1,0.7s_1+s_2)) \widetilde{Z}_2(s_1,0.7s_1+s_2) \mathrm{d}\nu(s),$$

and $(\lambda_n, \widetilde{\phi}_n)_{n=1}^{+\infty}$ are the solutions of the eigenvalue problem:

$$\int_{s\in\mathcal{S}} \sigma_2^2 r_2((s_1,0.7s_1+s_2),(u_1,0.7u_1+u_2)) \widetilde{\phi}_n(u) \mathrm{d}\nu(u) = \lambda_n \widetilde{\phi}_n(s).$$

$(\widetilde{\phi}_n)_{n\geqslant 1}$ is an orthonormal basis of $L^2([0,1],\nu)$:

$$\int_{s\in\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{\phi}_m(s) \mathrm{d}\nu(s) = \delta_{nm}.$$

This kernel cannot be used just as it is in practice, because the solutions of the eigenvalue problem are not explicit in general and the sums are infinite. To make this kernel tractable, a wise choice of r_2 (the correlation kernel of the latent process $\widetilde{Z}_2$) is suggested, for which an explicit formula is known. If the covariance kernel of $\widetilde{Z}_2$ is of the form:

$$\sigma_2^2 r_2((x_1,x_2),(t_1,t_2)) = \sigma_2^2 r_{2,1}(x_1,t_1) r_{2,2}(x_2-0.7x_1, t_2-0.7t_1),$$

with $r_{2,1}$ and $r_{2,2}$ stationary kernels, and if Z_2 is a P process, then according to proposition 13, it is equal to:

$$\begin{aligned} &Z_2(x_1,x_2) = \widetilde{Z}_2(x_1,x_2) \\ &- \begin{pmatrix} r_{2,2}(x_2-0.7x_1,0) & r_{2,2}(x_2-0.7x_1,0.2) \end{pmatrix} \begin{pmatrix} r_{2,2}(0,0) & r_{2,2}(0,0.2) \\ r_{2,2}(0.2,0) & r_{2,2}(0.2,0.2) \end{pmatrix}^{-1} \begin{pmatrix} \widetilde{Z}_2(x_1,0.7x_1) \\ \widetilde{Z}_2(x_1,0.7x_1+0.2) \end{pmatrix}. \end{aligned}$$

It is a centered Gaussian process of covariance kernel $\sigma_2^2\rho_2$ with:

$$\begin{aligned} &\rho_2((x_1,x_2),(t_1,t_2)) = r_{2,1}(x_1,t_1)[r_{2,2}(x_2-0.7x_1,t_2-0.7t_1) \\ &- \begin{pmatrix} r_{2,2}(x_2-0.7x_1,0) & r_{2,2}(x_2-0.7x_1,0.2) \end{pmatrix} \begin{pmatrix} r_{2,2}(0,0) & r_{2,2}(0,0.2) \\ r_{2,2}(0.2,0) & r_{2,2}(0.2,0.2) \end{pmatrix}^{-1} \begin{pmatrix} r_{2,2}(t_2-0.7t_1,0) \\ r_{2,2}(t_2-0.7t_1,0.2) \end{pmatrix}]. \end{aligned}$$

6.1.4 Test cases

The seqGPR metamodel is compared to the classic kriging metamodel (called K_tot) in the two analytic test cases from section 4.2 in chapter 4. Two types of parametrization are proposed for the $(Z_n)_{n=1}^N$ in the seqGPR metamodel:

- Semi-robust: Z_1 with one covariance parameter θ_1^i for each input variable x_i, $i \in I_1 \cup I_2$, for all n in $[\![2, N-1]\!]$, Z_n with one covariance parameter α_n common to all components of $x_{I_1 \cup \cdots \cup I_{n-1}}$, and one covariance parameter θ_n^i for each component of $x_{I_n \cup I_{n+1}}$, and Z_N with one covariance parameter α_N common to all components of $x_{I_1 \cup \cdots \cup I_{N-1}}$, and one covariance parameter θ_N^i for each input x_i, $i \in I_N$):

$$\left\{ \begin{array}{rcl} \theta_1 & = & \left(\underbrace{\theta_1^1, \cdots, \theta_1^{d_1}}_{I_1}, \underbrace{\theta_1^{d_1+1}, \cdots, \theta_1^{d_1+d_2}}_{I_2} \right), \\ \theta_n & = & \left(\underbrace{\alpha_n, \cdots, \alpha_n}_{I_1 \cup \cdots \cup I_{n-1}}, \underbrace{\theta_n^1, \cdots, \theta_n^{d_n}}_{I_n}, \underbrace{\theta_n^{d_n+1}, \cdots, \theta_n^{d_n+d_{n+1}}}_{I_{n+1}} \right), \quad \forall n \in [\![2, N-1]\!], \\ \theta_N & = & \left(\underbrace{\alpha_N, \cdots, \alpha_N}_{I_1 \cup \cdots \cup I_{N-1}}, \underbrace{\theta_N^1, \cdots, \theta_N^{d_N}}_{I_N} \right). \end{array} \right.$$

- Robust: Z_1 with one covariance parameter θ_1^i for each variable x_i, $i \in I_1$, and one covariance parameter β_1 common to all variables x_i, $i \in I_2$, for all n in $[\![2, N-1]\!]$, Z_n with one covariance parameter α_n common to all components of $x_{I_1 \cup \cdots \cup I_{n-1}}$, one covariance parameter θ_n^i for each component of x_{I_n}, and one covariance parameter β_n common to all components of $x_{I_{n+1}}$, and Z_N with one covariance parameter α_N common to all components of $x_{I_1 \cup \cdots \cup I_{N-1}}$, and one covariance parameter for each variable x_i, $i \in I_N$.

$$\left\{ \begin{array}{rcl} \theta_1 & = & \left(\underbrace{\theta_1^1, \cdots, \theta_1^{d_1}}_{I_1}, \underbrace{\beta_1, \cdots, \beta_1}_{I_2} \right), \\ \theta_n & = & \left(\underbrace{\alpha_n, \cdots, \alpha_n}_{I_1 \cup \cdots \cup I_{n-1}}, \underbrace{\theta_n^1, \cdots, \theta_n^{d_n}}_{I_n}, \underbrace{\beta_n, \cdots, \beta_n}_{I_{n+1}} \right), \quad \forall n \in [\![2, N-1]\!], \\ \theta_N & = & \left(\underbrace{\alpha_N, \cdots, \alpha_N}_{I_1 \cup \cdots \cup I_{N-1}}, \underbrace{\theta_N^1, \cdots, \theta_N^{d_N}}_{I_N} \right). \end{array} \right.$$

The classic kriging metamodel, K_tot, is trained on the reunion of the designs $(\mathbb{X}_1^1, \boldsymbol{y_1^1})$, $(\mathbb{X}_1^2, \boldsymbol{y_1^2})$ and $(\mathbb{X}_2, \boldsymbol{y_2})$. The covariance kernels of K_tot, Z_1, and $\left(\widetilde{Z_n}\right)_{n=2}^N$ (the latent processes on which are built $(Z_n)_{n=2}^N$) are tensor product Matern $\frac{5}{2}$ kernels, defined in subsection 2.1.1 in chapter 2.

Test case from subsection 4.2.2 in chapter 4 The output is the same as in subsection 4.2.2 in chapter 4:

$$\left\{\begin{array}{lcl} f(x_1,x_2,x_3,x_4) & = & g_1(x_1,x_2)+g_2(x_1,x_2,x_{3,4}) \\ \text{with} & & \\ g_1(x_1,x_2) & = & \left[4-2.1(4x_1-2)^2+\frac{(4x_1-2)^4}{3}\right](4x_1-2)^2 \\ & + & (4x_1-2)(2x_2-1)+\left[-4+4(2x_2-1)^2\right](2x_2-1)^2, \\ g_2(x_1,x_2,x_3,x_4) & = & 4\exp\left(-\|x-0.3\|^2\right). \end{array}\right.$$

The study is composed of two steps

- At step 1, three different restrictions are considered

$$\left\{\begin{array}{lcl} f_1^1(x_1,x_2) & = & f(x_1,x_2,0.4757155,0.7005368), \\ f_1^2(x_1,x_2) & = & f(x_1,x_2,0.2092920,0.2661060), \\ f_1^3(x_1,x_2) & = & f(x_1,x_2,0.9114971,0.3639307). \end{array}\right.$$

  The fixed values of (x_3,x_4) are chosen so that they form an LHS optimized for the generalized maximin criterion in $[0,1]^2$. The DoE is of the form:

$$\mathbb{X}_1=\begin{pmatrix} x_1^{(1)} & x_2^{(1)} & 0.4757155 & 0.7005368 & \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{(10)} & x_2^{(10)} & 0.4757155 & 0.7005368 & \\ x_1^{(1)} & x_2^{(1)} & 0.2092920 & 0.2661060 & \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{(10)} & x_2^{(10)} & 0.2092920 & 0.2661060 & \\ x_1^{(1)} & x_2^{(1)} & 0.9114971 & 0.3639307 & \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{(10)} & x_2^{(10)} & 0.9114971 & 0.3639307 & \end{pmatrix}.$$

  The same values of (x_1,x_2) are taken in the subspaces. They form an LHS optimized for the generalized maximin criterion in $[0,1]^2$.

- At step 2, new simulations are launched at points $\mathbb{X}_2$, a design of size 20 in dimension 4. The last two variables (x_3,x_4) are now released. $\mathbb{X}_2$ is an LHS, independent from $\mathbb{X}_1$, also optimized for the generalized maximin criterion.

Table 6.1 shows RMSE median and interquartile range of the different methods computed on a Sobol sequence of size 10000 used as a test set. The results are computed for 100 DoE's. The two versions of the seqGPR metamodel have worse performances than K_tot (higher median and wider interquartile interval). No type of parametrization stands out from the other. One explanation for this counter performance of the method is that it is more complex than in the previous case with one subspace at each step. The input spaces of the $(Z_n)_{n=1}^N$ are in higher dimension and the initial philosophy that consists in explaining a maximum of the data thanks to the small dimension is lost.

Test case from subsection 4.2.3 in chapter 4 The output is the same as in subsection 4.2.3 in chapter 4:

$$f \;:\; \left|\begin{array}{lcl} [-3,3]^{15} & \to & \mathbb{R} \\ x & \mapsto & a_1'x+a_2'\sin x+a_3'\cos x+x'Mx. \end{array}\right.$$

	K_tot	Semi-robust	Robust
Median ($\cdot 10^{-1}$)	**51.131**	55.499	55.529
Interquartile range ($\cdot 10^{-1}$)	**16.037**	23.102	22.741

Table 6.1: Performance of the metamodels for the 100 studies made on the analytic 4D test case. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$). The best performances are in bold.

	45pts-45pts/45pts			45pts-45pts/75pts			90pts-90pts/150pts		
	K	S	R	K	S	R	K	S	R
Median ($\cdot 10^{-4}$)	**62.014**	62.464	62.527	**39.287**	39.645	39.634	**7.605**	7.622	7.620
Interquartile range ($\cdot 10^{-4}$)	7.025	6.343	**6.342**	**6.709**	6.879	6.939	**0.594**	0.599	0.599

Table 6.2: Performance of the metamodels for the 100 studies made on the analytic 15D test case. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$). K denotes K_tot, S denotes the semi-robust version of seqGPR, and R denotes the robust version of seqGPR. 45pts-45pts/45pts means that $\mathbb{X}_1$ is of size 90 (with 45 points in the first subspace and 45 points in the second subspace), and that $\mathbb{X}_2$ is of size 45. 45pts-45pts/75pts means that $\mathbb{X}_1$ is of size 90 (with 45 points in the first subspace and 45 points in the second subspace), and that $\mathbb{X}_2$ is of size 75. 90pts-90pts/150pts means that $\mathbb{X}_1$ is of size 180 (with 90 points in the first subspace and 90 points in the second subspace), and that $\mathbb{X}_2$ is of size 150. The best performances are in bold.

The inputs are rescaled in $[0, 1]$ and are rearranged by decreasing order of Sobol index, but for the sake of simplicity, the resulting output is still denoted by f. The study is composed of two steps:

- At step 1, two restrictions are considered:

$$\begin{cases} f_1(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9) &= f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, 0, 0, 0, 0, 0, 0), \\ f_2(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9) &= f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, 1, 1, 1, 1, 1, 1). \end{cases}$$

  A DoE $\mathbb{X}_1$ is generated, with the same values of $(x_1, \cdots, x_9)$ taken in each subspace, such that they form an LHS optimized for the generalized maximin criterion.

- At step 2, the last variables $(x_{10}, \cdots, x_{15})$ are released. New simulations are launched at points $\mathbb{X}_2 \subset [0, 1]^{15}$. $\mathbb{X}_2$ is an LHS, independent from $\mathbb{X}_1$, also optimized for the generalized maximin criterion.

The RMSE median and interquartile range, computed on a Sobol sequence of size 10000 used as a test set, are given for 100 different DoE's $(\mathbb{X}^1, \boldsymbol{y}^1)$ and $(\mathbb{X}^2, \boldsymbol{y}^2)$ in table 6.2. The performances of the metamodels increase with the size of the training sample, but the comparison between the metamodels is the same for each training sample. Again, the same conclusion can be drawn: the two versions of the seqGPR metamodel have worse performances than K_tot (higher median and wider interquartile interval).

6.2 Enrichment

In this section, the non-nested designs are used: for all n in $[\![1, N]\!]$, the $x_{I_1 \cup \cdots \cup I_n}$ part of $\mathbb{X}_n$, $\cdots$, $\mathbb{X}_N$ have no points in common (see section 5.2 in chapter 5). An enrichment method for the DoE $\mathbb{X}_N$ of the seqGPR metamodel is suggested. The proposed optimization problem and the associated enrichment method are detailed. The method is then tested on analytic test cases.

6.2.1 Process of enrichment

The following optimization problem (derived from the one of [Le Gratiet, 2013b] described in subsection 2.1.5 in chapter 2) is chosen to seek an interesting point to add to the training sample

$$\begin{cases} \max\limits_{x \in \mathcal{X}} \ \hat{v}(x) \cdot \left[1 + \sum\limits_{i=1}^{n_1+\ldots+n_N} \frac{(y_i - \hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})} \mathbf{1}_{x \in V_i} \right], \\ \text{under the constraints:} \ \left| \begin{array}{l} dist(x_{I_1}, \widetilde{\mathbb{X}}_1) > u, \\ dist(x_{I_n}, \grave{x}_{I_n}) > v_n, \ \forall n \in [\![2, N]\!] \end{array} \right. \end{cases} \tag{6.5}$$

The thresholds u and $(v_n)_{n=2}^N$ are user defined. The constraints aim at keeping the invertibility property of the $\left(\widetilde{\mathbb{X}}_n\right)_{n=1}^N$ (so that $\left(\rho_n\left(\widetilde{\mathbb{X}}_n, \widetilde{\mathbb{X}}_n\right)\right)_{n=1}^N$ are invertible). The constraint $dist(x_{I_1}, \widetilde{\mathbb{X}}_1) > u$ also implies that $dist(x_{I_1 \cup \cdots \cup x_{I_n}}, \widetilde{\mathbb{X}}_n)$ is sufficiently big for $n \geqslant 2$.

The goal is to add a batch of points of size L (specified by the user) to the design $\mathbb{X}_N$, by solving the previously defined optimization problem. The enrichment algorithm is described below. It requires DoE's $(\mathbb{X}_1, \boldsymbol{y_1})$, ..., $(\mathbb{X}_N, \boldsymbol{y_N})$, and a seqGPR metamodel trained on these samples.

1. Compute the LOO-CV errors $\frac{(y_i - \hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})}$ of the seqGPR metamodel. These will not be updated during the algorithm.

2. Generate a sobol sequence of size 1000000 in the whole input space $[0, 1]^{d_1+\cdots+d_N}$. The batch of enrichment points will be choosen among these points.

3. Keep only the points verifying the constraints.

4. Choose the point optimal for the criterion.

5. Add the point to $\mathbb{X}_N$ and update the prediction variance $\hat{v}$ of the seqGPR metamodel, but not the LOO-CV errors.

6. Start again to phase 3, until the size L has been reached.

6.2.2 Example in 2D

In the example in 2D from subsection 3.1.2 in chapter 3, the metamodel seqGPR has been built on the non-nested samples:

$$\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}, \ \mathbb{X}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} \\ \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} \end{pmatrix},$$

with the associated output values $\boldsymbol{y_1}$ and $\boldsymbol{y_2}$. The goal is to add new relevant points to $\mathbb{X}_2$ to improve the metamodel accuracy. An optimization problem is defined which has to be solved by the new point. This optimization problem involves an objective function to maximize defined by [Le Gratiet, 2013b] (see subsection 2.1.5 in chapter 2):

$$\tau(x) = \hat{v}(x) \left[1 + \underbrace{\sum_{i=1}^{n_1} \frac{(y_i - \hat{y}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2))^2}{\hat{v}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2)} \mathbf{1}_{V_i}(x)}_{=\tau_1(x)} + \underbrace{\sum_{i=n_1+1}^{n_1+n_2} \frac{(y_i - \hat{y}_{-i}(x_1^{(i)}, x_2^{(i)}))^2}{\hat{v}_{-i}(x_1^{(i)}, x_2^{(i)})} \mathbf{1}_{x \in V_i}}_{=\tau_2(x)} \right]$$

This enrichment criterion is the product of two terms. The first one is the prediction variance of the seqGPR metamodel, it is a spatial criterion that favours points far from the designs $\mathbb{X}_1$ and $\mathbb{X}_2$. The second one takes into account the prediction error of the seqGPR metamodel by involving an LOO-CV criterion. τ_1 (resp. τ_2) is an LOO-CV criterion applied to the design $\mathbb{X}_1$ (resp. $\mathbb{X}_2$). $\hat{y}_{-i}$ and $\hat{v}_{-i}$ are the prediction mean and variance if the ith training point is not used in the metamodel fitting, and $y_i - \hat{y}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2)$ (resp. $y_i - \hat{y}_{-i}(x_1^{(i)}, x_2^{(i)})$) is the associated prediction error. V_i is the Voronoï cell of the ith training point. The Voronoï cells $(V_i)_{i=1}^{n_1+n_2}$ form a partition of the input space, for each i, V_i contains the points that are closer to the ith training point than the other training points. The prediction variance is an estimation of the prediction error and the criterion τ acts as a corrected estimation of the prediction error. Indeed, in areas where the prediction error is supposedly underestimated, i.e. in V_i such that $(y_i - \hat{y}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2))^2 >> \hat{v}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2)$ (resp. $(y_i - \hat{y}_{-i}(x_1^{(i)}, x_2^{(i)}))^2 >> \hat{v}_{-i}(x_1^{(i)}, x_2^{(i)})$), the second factor contributes to increasing the value of τ.

Let $x = (x_1, x_2)$ denote the new point. The enriched designs are now denoted by $\mathbb{X}_2^+$, $\overset{\sim}{\mathbb{X}}_1^+$:

$$\overset{\sim}{\mathbb{X}}_1^+ = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1+n_2)} \\ x_1 \end{pmatrix}, \ \mathbb{X}_2^+ = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} \\ \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} \\ x_1 & x_2 \end{pmatrix}.$$

Some regions of the input space are ineligible to contain x in order to maintain the invertibility of $\rho_1(\overset{\sim}{\mathbb{X}}_1^+, \overset{\sim}{\mathbb{X}}_1^+)$ and $\rho_2(\mathbb{X}_2^+, \mathbb{X}_2^+)$ (the covariance matrices involved in the EM algorithm, see subsection 4.1.2).

- In order for $\rho_1(\overset{\sim}{\mathbb{X}}_1^+, \overset{\sim}{\mathbb{X}}_1^+)$ to be well-conditioned, x_1 cannot already belong to $\overset{\sim}{\mathbb{X}}_1$. The related constraint for the optimization problem is $dist(x_1, \overset{\sim}{\mathbb{X}}_1) > u_1$ with $u_1 > 0$ a user-defined scalar and:

$$dist(x_1, \overset{\sim}{\mathbb{X}}_1) = \min_{1 \leqslant i \leqslant n_1+n_2} \left| x_1 - x_1^{(i)} \right|.$$

- In order for $\rho_2(\mathbb{X}_2^+, \mathbb{X}_2^+)$ to be well-conditioned, $x = (x_1, x_2)$ cannot already belong to $\mathbb{X}_2$. The related constraint for the optimization problem is $dist(x, \mathbb{X}_2) > u_2$ with $u_2 > 0$ a user-defind scalar and:

$$dist(x, \mathbb{X}_2) = \min_{n_1+1 \leqslant i \leqslant n_1+n_2} \sqrt{(x_1 - x_1^{(i)})^2 + (x_2 - x_2^{(i)})^2}.$$

  In fact, the first constraint implies this one, which is not kept.

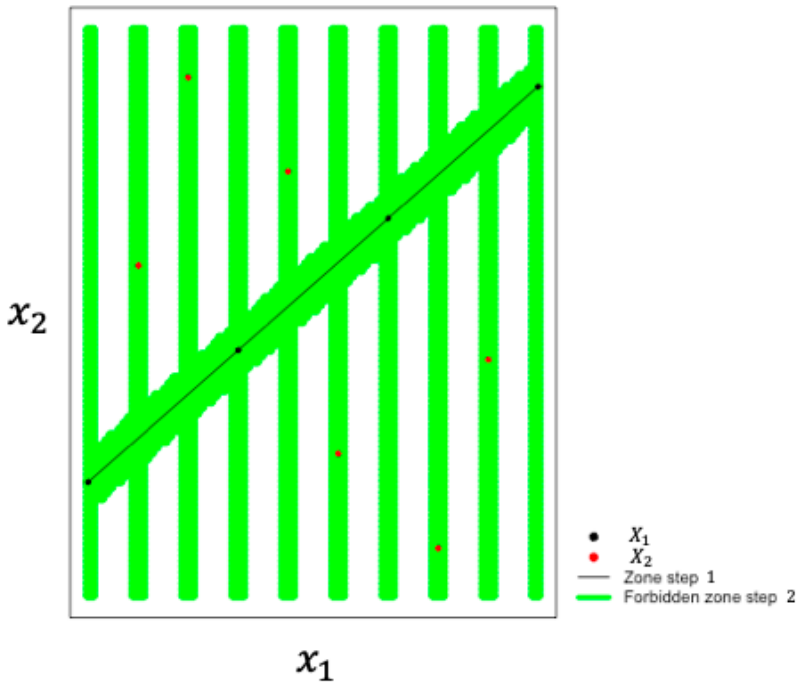
- In order for $\rho_2(\mathbb{X}_2^+, \mathbb{X}_2^+)$ to be well-conditioned, x must also not belong to the nullity zone of Z_2, i.e. x_2 must be different from $0.7x_1 + 0.2$. The related constraint for the optimization problem is $dist(x_2, 0.7x_1 + 0.2) > v$ with $v > 0$ a user-defined scalar and:

$$dist(x_2, 0.7x_1 + 0.2) = |x_2 - 0.7x_1 - 0.2|\,.$$

To summarize, the optimization problem contains two constraints:

$$\begin{cases} dist(x_1, \widetilde{\mathbb{X}}_1) > u, \\ dist(x_2, 0.7x_1 + 0.2) > v. \end{cases}$$

The areas which are not eligible are illustrated in green in the graph below:



The goal may be to add several points at a time. In this case, the enrichment algorithm is described below.

1. Compute the LOO-CV errors $\frac{(y_i - \hat{y}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2))^2}{\hat{v}_{-i}(x_1^{(i)}, 0.7x_1^{(i)} + 0.2)}$ $(1 \leqslant i \leqslant n_1)$ and $\frac{(y_i - \hat{y}_{-i}(x_1^{(i)}, x_2^{(i)}))^2}{\hat{v}_{-i}(x_1^{(i)}, x_2^{(i)})}$ $(n_1 + 1 \leqslant i \leqslant n_1 + n_2)$ of the seqGPR metamodel. These will not be updated during the algorithm.

2. Generate a sobol sequence of size 1000000 in the whole input space $[0, 1]^2$. The batch of enrichment points will be choosen among these points.

3. Keep only the points verifying the constraints.

4. Choose the point optimal for the criterion.

5. Add the point to $\mathbb{X}_2$ and update the prediction variance $\hat{v}$ of the seqGPR metamodel, but not the LOO-CV errors.

6. Start again to phase 3, until the required number of points is reached.

6.2.3 Test cases

The tests are done with the seqGPR metamodel with the Red process and robust kernel (see section 4.2 in chapter 4). On the different test cases, the initial metamodel (I) is compared to the one enriched with 10 points. Three methods are tried to seek the enrichment points:

- C (criterion): the method described in the previous subsection 6.2.1.
- V (var): same method as C but replacing the criterion to optimize by the prediction variance alone $\hat{v}$ (without the LOO-CV errors).
- R (random): ten points taken randomly among the points of the sobol sequence of size 10000.

The test cases are the same than in section 4.2 in chapter 4. The initial seqGPR model is the one computed during these test cases.

Test case from subsection 4.2.2 in chapter 4 It is the exact same test case with the same step definition. The enrichment method is tested on the cases 10pts/20pts ($\mathbb{X}_1$ of size 10 and $\mathbb{X}_2$ of size 20) and 20pts/40pts ($\mathbb{X}_1$ of size 20 and $\mathbb{X}_2$ of size 40) with non-nested designs. The results are shown on table 6.3. In the case 10ts/20pts, only the R enrichment approach improves the accuracy of the metamodel, the two others disturb it (same median but higher interquartile range than the initial metamodel). However, in the case 20pts/40pts, the enrichments V and C, using the variance or the criterion of subsection 6.2.1 perform better, and especially the method C which is the best in this case. One conclusion is that the method C seems efficient only if the metamodel is sufficiently accurate. The enrichment can be divided in two parts, one with random points added until the metamodel has reached a certain accuracy level, and the following part with points added using the method C. One perspective is then to determine when to switch from one part to the other and when to stop the enrichment.

Test case from subsection 4.2.3 in chapter 4 As a recall, $\mathbb{X}_1$ is of size 15, $\mathbb{X}_2$ of size 45 and $\mathbb{X}_3$ of size 75 and the designs are not nested. The results are shown on table 6.4. This time the median of V and C are equivalent and C has a smaller interquartile range. Therefore, it seems to be the best enrichment strategy, but not by much.

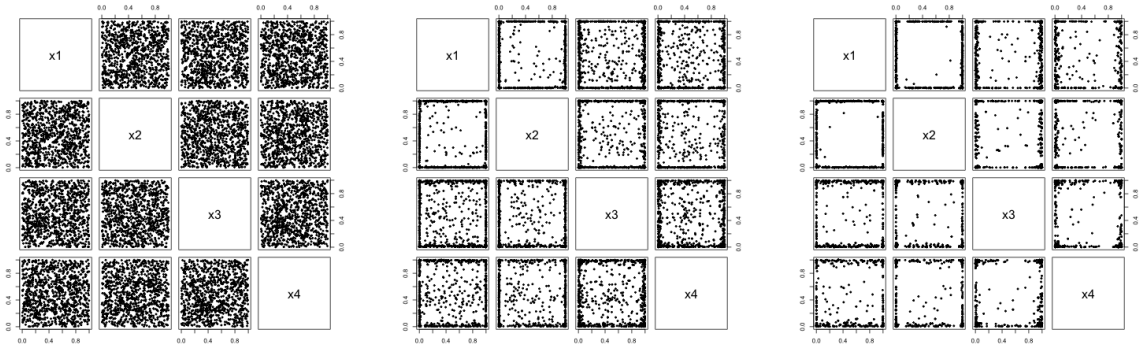
Location of the enrichment points Figure 6.2 (resp. figure 6.3) shows the locations of the enrichment points for the analytic test case in 4D described in subsection 4.2.2 in chapter 4 with initial DoE's $\mathbb{X}_1$ of size 10 and $\mathbb{X}_2$ of size 20 (resp. $\mathbb{X}_1$ of size 20 and $\mathbb{X}_2$ of size 40). Figure 6.4 shows the locations of the enrichment points for the analytic test case in 15D described in subsection 4.2.3 in chapter 4. The same conclusion can be drawn from all the figures. The enrichment points choosen at random are unsurprisingly uniformly spread in

	10pts/20pts				20pts/40pts			
	I	R	V	C	I	R	V	C
Median ($\cdot 10^{-2}$)	42.108	**31.915**	40.079	38.041	7.292	4.586	4.044	**3.916**
Interquartile range ($\cdot 10^{-2}$)	9.588	**5.870**	13.331	15.651	2.100	1.360	1.050	**0.847**

Table 6.3: Performance of the metamodels for the 100 studies made on the analytic 4D test case. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$). I denotes the initial metamodel seqGPR, R denotes the metamodel enriched with random points, V denotes the metamodel enriched by maximum variance, and C denotes the metamodel enriched with maximum criterion (defined in subsection 6.2.1). 10pts/20pts means that $\mathbb{X}_1$ is of size 10 and that $\mathbb{X}_2$ is of size 20. 20pts/40pts means that $\mathbb{X}_1$ is of size 20 and that $\mathbb{X}_2$ is of size 40. The best performances are in bold.

	I	R	V	C
Median ($\cdot 10^{-4}$)	36.903	29.151	**26.763**	**27.208**
Interquartile range ($\cdot 10^{-4}$)	6.570	6.947	6.693	**5.594**

Table 6.4: Performance of the metamodels for the 100 studies made on the analytic 15D test case. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$). I denotes the initial metamodel seqGPR, R denotes the metamodel enriched with random points, V denotes the metamodel enriched by maximum variance, and C denotes the metamodel enriched with maximum criterion (defined in subsection 6.2.1). The best performances are in bold.



(a) Random (b) criterion (c) Variance

Figure 6.2: Location of the enrichment points of the 100 studies on the analytic 4D test case using the seqGPR metamodel based on the training samples $\mathbb{X}_1$ in 2D of size 10 and $\mathbb{X}_2$ in 4D of size 20. The enrichment points in left panel are choosen at random, those on middle panel are choosen using the criterion defined in (6.5), and those on the right panel are choosen using the prediction variance.

the input space. The enrichment points choosen by maximum prediction variance are mostly located on the sides and corners of the input space (and especially in the corners in the 4D analytic test case). Those choosen by the criterion defined in (6.5) are also located on the sides of the input space for most of them but there are more points within the input space.

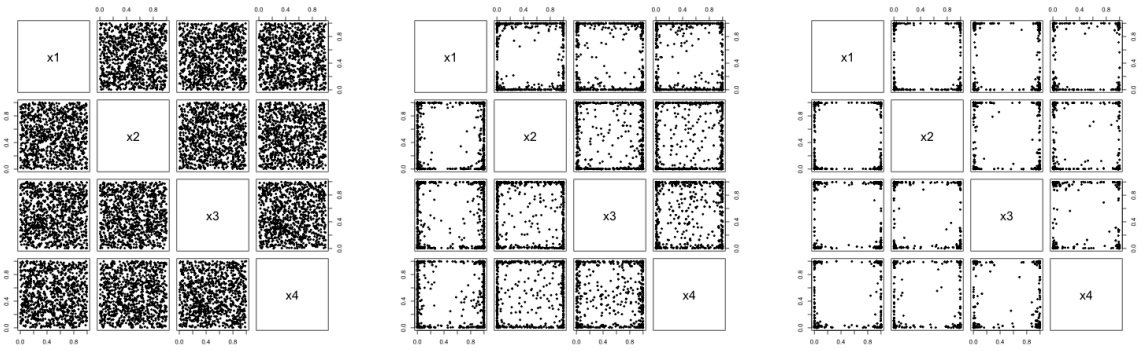
6.3 Conclusion

This chapter tackles the case where multiple subspaces are explored at each step instead of one. An adaptation of the seqGPR metamodel is suggested and a new candidate for Z_n (n in $[\![2, N]\!]$) is introduced, derived from the P process. The method is compared to a classic kriging metamodel on the same test cases as in section 4.2 in chapter 4, but this time, the seqGPR metamodel is not better than K_tot. This adapted version of the seqGPR metamodel may be too complex, the advantage of Z_n (with n small) learning f in small dimension is lost.

This chapter also deals with the issue of enriching the DoE's of the seqGPR metamodel. A strategy is proposed to add a batch of points verifying some constraints proper to the seqGPR metamodel and optimal for a criterion found in the literature. The method is compared to other enrichment strategies on the test cases of section 4.2 in chapter 4. It is not as convincing as expected. A study of the enrichment points location shows that they are mostly located in the sides of the input space, even if this trend is less pronounced than for the maximum prediction variance enrichment. As an enrichment within the input space is more interesting (see [Henner et al., 2019] in appendix 9.5.1), a new enrichment method can be proposed, either by applying the criteria defined in (6.5) but adding to the constraints a minimal distance to the sides, or by defining a new criteria derived from the one defined in (6.5), taking into account the distance to the sides:

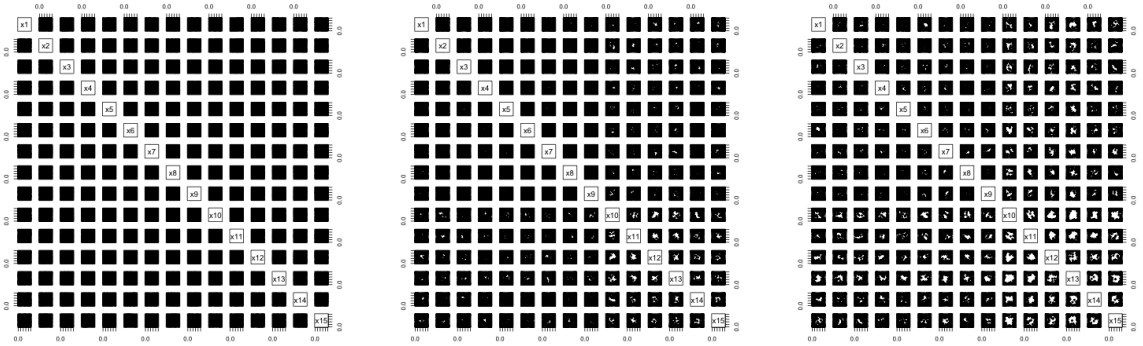
$$\hat{v}(x) \left[1 + \sum_{i=1}^{n_1+\ldots+n_N} \frac{(y_i - \hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})} \mathbf{1}_{x \in V_i}\right] \left[1 + \sum_{i=1}^{d_1+\ldots+d_N} -x_i(x_i - 1)\right].$$

The new factor is maximal when all x components are equal to 0.5 (see figure 6.5).



(a) Random (b) Criterion (c) Variance

Figure 6.3: Location of the enrichment points of the 100 studies on the analytic 4D test case using the seqGPR metamodel based on the training samples $\mathbb{X}_1$ in 2D of size 20 and $\mathbb{X}_2$ in 4D of size 40. The enrichment points in left panel are choosen at random, those on middle panel are choosen using the criterion defined in (6.5), and those on the right panel are choosen using the prediction variance.



(a) Random (b) criterion (c) Variance

Figure 6.4: Location of the enrichment points of the 100 studies on the analytic 15D test case using the seqGPR metamodel based on the training samples $\mathbb{X}_1$ in 3D of size 15, $\mathbb{X}_2$ in 9D of size 45, and $\mathbb{X}_3$ in 15D of size 75. The enrichment points in left panel are choosen at random, those on middle panel are choosen using the criterion defined in (6.5), and those on the right panel are choosen using the prediction variance.

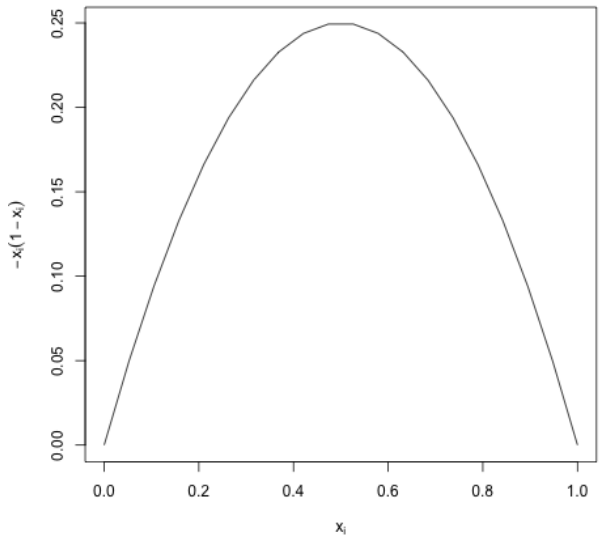


Figure 6.5: Visualization of $-x_i(x_i - 1)$, the term associated to the component x_i of the new factor.

Chapter 7

Conclusions and perspectives

7.1 Conclusions

The goal of this thesis is to build a metamodel defined on nested subspaces of increasing dimension, i.e. some input variables are fixed and then released progressively. The work first consists in the definition of a method, called seqGPR (sequential Gaussian process regression). It consists in creating a kriging metamodel with a particular kernel on the reunion of all DoE's. A probabilistic framework is defined in which the kernel (which is unstationary) is the covariance kernel of a Gaussian process (modeling the output at the last step) defined recursively. At a given step, the process modeling the output is equal to the sum of the process modeling the output at the previous step and an independent correction term.

The correction term is a Gaussian process null on a continuous part of its definition space. One issue raised by the probabilistic framework is then to define such a process both verifying the nullity property and with a tractable kernel. Three candidates are proposed : Psi (latent process multiplied a deterministic function null on the concerned subspace), Red (latent process minus its value on the subspace), and P (latent process minus its conditional expectation on the subspace). The P process comes from the literature, and a work is done in the thesis to make it tractable. The Psi process is not retained, as it has the drawback of generating disturbed paths and as it has more parameters to estimate.

Another issue raised by the definition of the seqGPR metamodel is the estimation of its parameters. The choice is set on the maximum likelihood estimation and two simplifications are made : decoupling of the likelihood in the case of nested designs, use of an EM (Expectation-Maximization) algorithm when the designs are non-nested.

The metamodel seqGPR is tested on two analytic test cases and an industrial one. Different types of parametrization are tested and the method is compared to a classic kriging metamodel. Results show that non-nested designs are more performant than nested designs, the method seqGPR is better than the classic kriging metamodel, the best metamodel is the seqGPR with a Red robust kernel.

Methods are detailed to generate nested or non-nested designs in case they are not furnished by the user. These methods enable a good conditioning of the matrices to invert in the parameter estimation and a good space filling property of the designs.

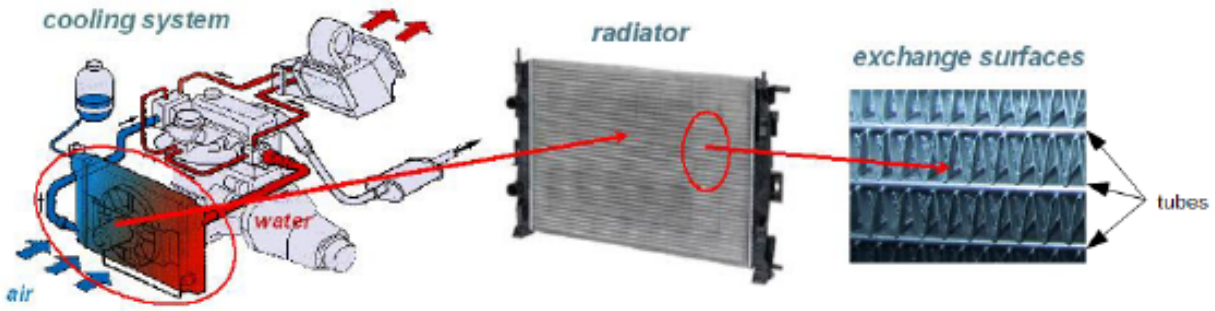


Figure 7.1: Diagrams of the car engine cooling system and its radiator. [Valeo,]

7.2 Perspectives

7.2.1 Multi-conditioning

The first perspective is to adapt the seqGPR metamodel in the case of multiple subspaces at each step. A first suggestion of adaptation is made, involving a generalization of the P process. The tests on analytical test cases show similar accuracy between the adapted seqGPR metamodel and the classic kriging metamodel. Future work will consist in finding alternative ways of defining the seqGPR metamodel in this context.

7.2.2 Enrichment

The second perspective is to enrich the training samples of the seqGPR metamodel to improve its accuracy. A strategy is suggested to enrich the training sample with a batch of points of size defined by the user. This batch of points must verify a certain optimization problem. This enrichment strategy does not give significantly better results than more classical ones on the two analytical test cases it was tested on. Future work will consist in studying alternative enrichment criteria like the one proposed in the conclusion of chapter 6 that involves a factor taking into account the distance to the input space sides. A completely new optimization problem can also be searched, mixing mathematical and physical knowledge.

7.2.3 Categorical variables

The next step is then to adapt the seqGPR metamodel to a test case mixing continuous and categorical variables. Such a test case is already studied in Valeo. As for the fan system (see section 1.1 in chapter 1), the newly studied system is also part of the car engine cooling system (see figure 7.1). Via this system, a cooling fluid circulates in the engine to decrease its temperature. On the left part of figure 7.1, the hot fluid getting out of the engine is represented in red. The fluid itself is then cooled down in the radiator via the fan system which projects air on it. The cold fluid getting out of the radiator is represented in blue. The middle part of the figure represents the radiator. It is composed of a vertical stacking of, alternately, fluid circulation tubes and air inlets, as shown in the right part.
The studied system is part of an air inlet (see figure 7.2). The air inlet plays the role of a turbulence generator, to favour heat exchange. The air inlet is constituted of a periodic horizontal juxtaposition of an elementary pattern. The middle part of the figure represents one air inlet with a circulation tube. The system studied in this subsection is this elementary pattern, called fin system, whose diagram is represented on the right part of the figure.

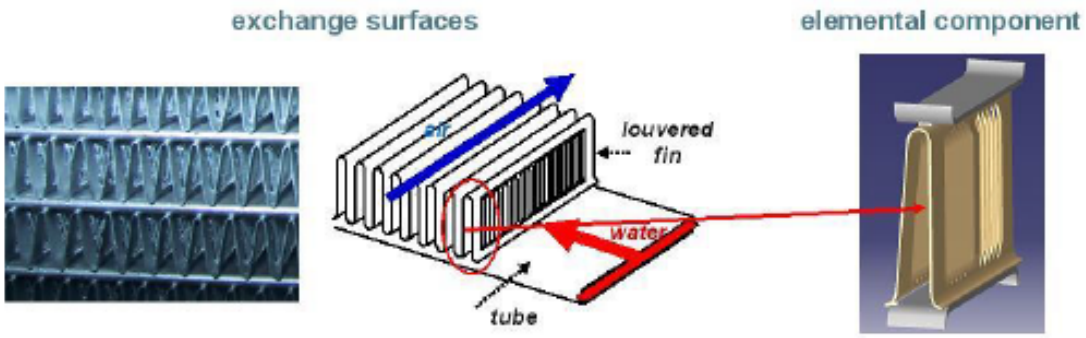


Figure 7.2: Diagrams of the exchange surface air-fluid and of the elementary pattern. [Valeo,]

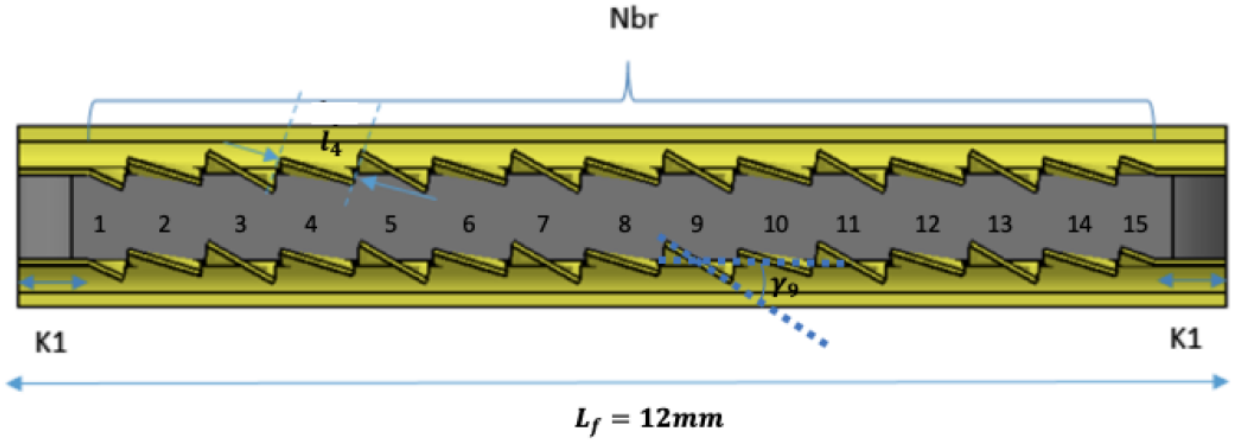
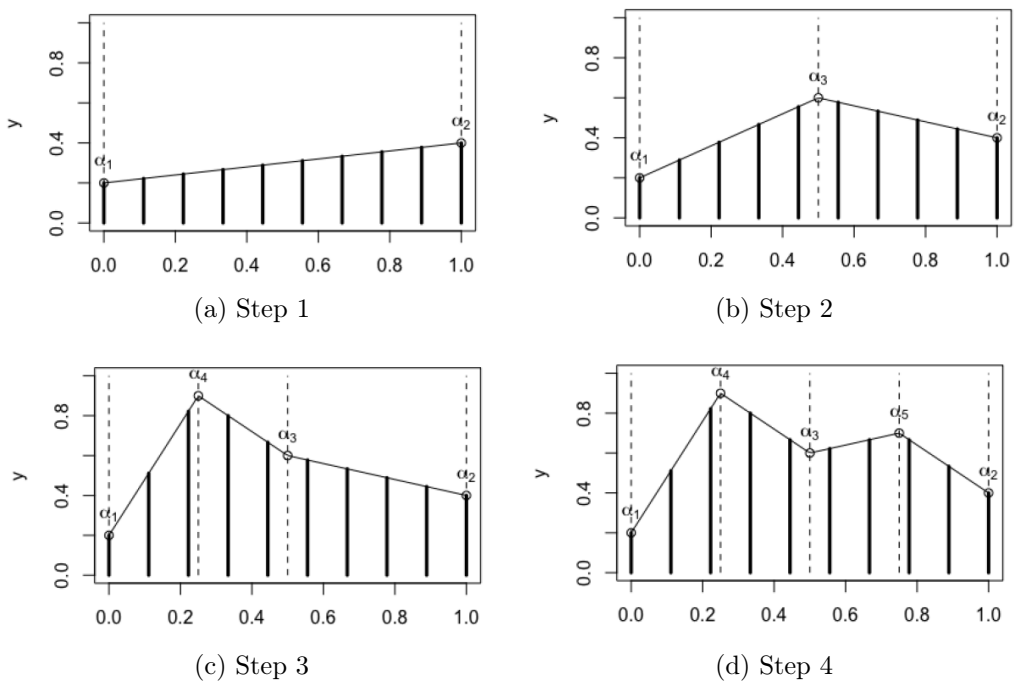


Figure 7.3: Diagram of the fin system and its parameters. [Valeo,]

Figure 7.3 shows a view from below of the elementary component. On the diagram, the air circulates from left to right and the fluid from top to bottom. The number of louvers of the fin system, denoted by Nbr, takes its values in the set $\{10, 12, 14, 16, 18, 20, 22\}$. It is a categorical variable with 7 levels. The flat area on the sides of the louvers, has its length K_1 varying between 0.5 and 1mm. The louvers are distributed on two parallel rows. The two rows are identical to each other. The fin length is $Lf = 12$mm. In the direction of air propagation, the total length granted to the louvers on each row is $Lf - 2K_1$. The louvers are numbered from 1 to Nbr. The ith louver has a rotation angle γ_i with respect to the air direction varying between 17 and 30 degrees. For the moment, all the louver lengths l_i are fixed at a nominal value.
The angles cannot be taken as input variables as they are not the same for different values of Nbr. Valeo experts rather choose to model the angle distribution by a parameterized curve, piecewise affine, as shown in figure 7.4d. The curve is defined on $[0, 1]$. Its has 5 parameters: α_1, α_2, α_3, α_4, α_5, which are respectively the curve values at 0, 1, 0.5, 0.25, and 0.75. Then the angle values $\gamma_1, \cdots, \gamma_{Nbr}$ are determined by taking the curve values at evenly distributed points in $[0, 1]$.
The input variables of the model are: Nbr, K_1, α_1, α_2, α_3, α_4, α_5. The output of interest is the pressure lost (in Pa). A study of 4 steps is done on the fin system:

- At step 1 , the values of α_3, α_4, and α_5 are respectively fixed to $\alpha'_3 = \frac{\alpha'_4+\alpha'_5}{2} = \frac{\alpha_1+\alpha_2}{2}$,



(a) Step 1

(b) Step 2

(c) Step 3

(d) Step 4

Figure 7.4: Curves modeling the angle distribution at the different steps. The angles values are shown by the black vertical lines.

$\alpha'_4 = \frac{\alpha_1+\alpha'_3}{2} = \frac{3\alpha_1+\alpha_2}{4}$, $\alpha'_5 = \frac{\alpha'_3+\alpha_2}{2} = \frac{\alpha_1+3\alpha_2}{4}$ (see figure 7.4a). The variables Nbr, K_1, α_1, α_2 are free.

- At step 2, α_3 is released. α_4 and α_5 are still fixed to $\alpha'_4 = \frac{\alpha_1+\alpha_3}{2}$ and $\alpha'_5 = \frac{\alpha_3+\alpha_2}{2}$ (see figure 7.4b). The variables Nbr, K_1, α_1, α_2, α_3 are free.
- At step 3, α_4 is released. α_5 is still fixed to $\alpha'_5 = \frac{\alpha_3+\alpha_2}{2}$ (see figure 7.4c). The variables Nbr, K_1, α_1, α_2, α_3, α_4 are free.
- At step 4, α_5 is released (see figure 7.4d). All variables are free.

Thus one perspective is to adapt the seqGPR metamodel to this test case which has the particularity of involving a categorical variable Nbr.

Chapter 8

Résumé en Français

Le but de cette thèse est d'approximer par un métamodèle une fonction $f(x_1, \cdots, x_d)$ dont on connait les valeurs en plusieurs plans d'entraînement $(\mathbb{X}_1, \boldsymbol{y_1})$, $\cdots$, $(\mathbb{X}_N, \boldsymbol{y_N})$, qui ont été construits au cours d'une étude composée de N étapes. Le plan $\mathbb{X}_n \subset [0,1]^{d_1+\cdots+d_n}$, généré à l'étape $n \in [\![1, N]\!]$, est localisé dans une zone précise de l'espace des entrées. Il contient les valeurs des variables $(x_{I_1}, \cdots, x_{I_n})$ tandis que les variables $(x_{I_{n+1}}, \cdots, x_{I_N})$ sont fixées aux valeurs $(\grave{x}_{I_{n+1}}, \cdots, \grave{x}_{I_N})$ (ces valeurs sont imposées, elles peuvent être constantes ou dépendre de façon déterministe des variables libres). Le vecteur I_n contient les indices des variables libérées à l'étape n. Pour prendre en compte cette distribution particulière des plans d'expérience, la modélisation suivante est proposée. La fonction f est supposée être la réalisation d'un certain processus Gaussien Y_N défini récursivement par:

$$\left\{ \begin{array}{lcl} Y_1(x_{I_1}) & = & m + Z_1(x_{I_1}), \\ Y_n(x_{I_1}, \cdots, x_{I_{n-1}}, x_{I_n}) & = & Y_{n-1}(x_{I_1}, \cdots, x_{I_{n-1}}) + Z_n(x_{I_1}, \cdots, x_{I_{n-1}}, x_{I_n}), \quad \forall n \in [\![2, N]\!]. \end{array} \right.$$

Le processus Z_n $(n \geqslant 2)$ est un processus correctif captant l'information apportée par les variables nouvellement libérées. Il est nul sur la partie de l'espace correspondant à l'étape $n-1$ afin que le processus Y_n coincide avec le processus Y_{n-1} sur cette zone:

$$Z_n(x_{I_1}, \cdots, x_{I_{n-1}}, \grave{x}_{I_n}) = 0.$$

Le métamodèle seqGPR utilise les formules habituelles de moyenne et variance de prédiction, qui sont les moments du processus conditionnel $[Y_N(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, \cdots, Y_N(\mathbb{X}_N) = \boldsymbol{y_N}]$:

$$\left\{ \begin{array}{lcl} \hat{y}(x) & = & \mathbb{E}\left[Y_N(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, \cdots, Y_N(\mathbb{X}_N) = \boldsymbol{y_N}\right], \\ \hat{v}(x) & = & Var\left(Y_N(x) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, \cdots, Y_N(\mathbb{X}_N) = \boldsymbol{y_N}\right). \end{array} \right.$$

Trois questions se posent. Comment construire les processus $(Z_n)_{n=2}^N$? Comment construire les plans $(\mathbb{X}_n)_{n=1}^N$? Comment estimer les paramètres du modèle ?

8.1 Définition des processus correctifs $(Z_n)_{n=2}^N$

La première question à résoudre est de définir un tel processus Z_n Gaussien, vérifiant la propriété de nullité, et dont le noyau soit facilement calculable. Trois candidats sont proposés,

les processus Psi, Red et P définis par:

$$\begin{cases} Z_n^{Psi}(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) = \Psi(x_{I_n} - \grave{x}_{I_n})\widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}), \\ \\ Z_n^{Red}(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) = \widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) - \widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, \grave{x}_{I_n}), \\ \\ Z_n^{P}(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) = \widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) \\ -\mathbb{E}\left[\widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n})) \mid \widetilde{Z}_n(t_{I_1\cup\cdots\cup I_{n-1}}, \grave{t}_{I_n}),\ \forall t_{I_1\cup\cdots\cup I_{n-1}} \in [0,1]^{d_1+\cdots+d_{n-1}}\right]. \end{cases}$$

$\widetilde{Z}_n$ est un processus Gaussien dont le noyau de covariance est calculable directement. Il est appelé processus latent. C'est à partir de lui que sont construits les trois candidats. La fonction Ψ est définie par :

$$\Psi(t) = 1 - \exp\left(-\sum_{i=1}^{d_n} \frac{t_i}{2\delta_i}\right).$$

Elle s'annule en 0 et vaut 1 loin de 0. Enfin l'espérance intervenant dans la définition du processus P est la projection orthogonale de $\widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n})$ sur l'espace Gaussien engendré par les variables $\left(\widetilde{Z}_n(t_{I_1\cup\cdots\cup I_{n-1}}, \grave{t}_{I_n})\right)_{t_{I_1\cup\cdots\cup I_{n-1}}\in[0,1]^{d_1+\cdots+d_{n-1}}}$. Le processus P a une forme analytique facilement calculable numériquement dans le cas précis où le processus latent $\widetilde{Z}_n$ a un noyau de covariance de la forme :

$$k((x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}), (t_{I_1\cup\cdots\cup I_{n-1}}, t_{I_n})) = \sigma^2 r_1(x_{I_1\cup\cdots\cup I_{n-1}}, t_{I_1\cup\cdots\cup I_{n-1}}) r_2(x_{I_n} - \grave{x}_{I_n}, t_{I_n} - \grave{t}_{I_n}).$$

Dans ce cas, le processus P s'écrit :

$$Z_n^P(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) = \widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n}) - r_2(x_{I_n} - \grave{x}_{I_n}, 0)\widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, \grave{x}_{I_n}).$$

Le processus Psi est finalement abandonné car l'estimation de ses paramètres est plus compliquée que pour les deux autres.

8.2 Construction des plans $(\mathbb{X}_n)_{n=1}^N$

Deux types de familles de plans sont considérés : les plans imbriqués et les plans non imbriqués.

8.2.1 Les plans imbriqués

Le plan $\mathbb{X}_n$ ($n \geqslant 2$) est dit imbriqué dans $\mathbb{X}_{n-1}$ s'ils sont de la forme :

$$\begin{cases} \mathbb{X}_{n-1} &= \begin{pmatrix} x_{I_1\cup\cdots\cup I_{n-1}}^{(1)} \\ \cdots \\ x_{I_1\cup\cdots\cup I_{n-1}}^{(n_{n-1})} \end{pmatrix}, \\ \\ \mathbb{X}_n &= \begin{pmatrix} x_{I_1\cup\cdots\cup I_{n-1}}^{(i_1)} & x_{I_n}^{(i_1)} \\ \cdots & \\ x_{I_1\cup\cdots\cup I_{n-1}}^{(i_{n_n})} & x_{I_n}^{(i_{n_n})} \end{pmatrix}, \quad \text{with } \{i_1, \cdots, i_{n_n}\} \subset \{1, \cdots, n_{n-1}\}. \end{cases}$$

Le block de $\mathbb{X}_n$ correspondant aux variables $x_{I_1\cup\cdots\cup I_{n-1}}$ reprend les lignes de $\mathbb{X}_{n-1}$. Pour construire le plan $\mathbb{X}_n$ à partir du plan $\mathbb{X}_{n-1}$, un algorithme de recuit simulé est utilisé pour intervertir les lignes de la matrice :

$$\left[\begin{array}{c|ccc} x^{(1)}_{I_1\cup\cdots\cup I_{n-1}} & \frac{1}{2n_n} & \cdots & \frac{1}{2n_n} \\ \vdots & \vdots & & \vdots \\ x^{(k)}_{I_1\cup\cdots\cup I_{n-1}} & \frac{1}{2n_n}+\frac{k-1}{n_n} & \cdots & \frac{1}{2n_n}+\frac{k-1}{n_n} \\ \vdots & \vdots & & \vdots \\ x^{(n_n)}_{I_1\cup\cdots\cup I_{n-1}} & 1-\frac{1}{2n_n} & \cdots & 1-\frac{1}{2n_n} \\ \hline x^{(n_n+1)}_{I_1\cup\cdots\cup I_{n-1}} & -1 & \cdots & -1 \\ \vdots & \vdots & & \vdots \\ x^{(n_{n-1})}_{I_1\cup\cdots\cup I_{n-1}} & -1 & \cdots & -1 \end{array}\right].$$

Dans la partie gauche, les lignes $x^{(k)}_{I_1\cup\cdots\cup I_{n-1}}$ (k dans $[\![1, n_{n-1}]\!]$) doivent être interverties d'un block. Dans la partie droite, les intervertions se font à l'intérieur d'une même colonne. $\mathbb{X}_n$ est finalement égale à la matrice résultant de ce recuit simulé dont sont enlevées les lignes comportant des -1. Le recuit simulé se fait dans le but de minimiser la fonction objectif suivante :

$$\underbrace{\left(\sum_{\substack{x,t\in\mathbb{X}_n\\ x\neq t}}\left(\frac{1}{\|x-t\|^2_{[0,1]^{d_1+\cdots+d_n}}}\right)^{50}\right)^{\frac{1}{50}}}_{=\phi_1}+\underbrace{\left(\sum_{x\in\mathbb{X}_n}\left(\frac{1}{\|x_{I_n}-\grave{x}_{I_n}\|^2_{[0,1]^{d_n}}}\right)^{50}\right)^{\frac{1}{50}}}_{\phi_2},$$

où Φ_1 est un critère de maximin généralisé qui fait en sorte que les points soient bien répartis dans $[0,1]^{d_1+\cdots+d_n}$, et Φ_2 impose que $\mathbb{X}_n$ soit éloigné de la zone d'annulation de Z_n.

8.2.2 Les plans non imbriqués

Dans le cas des plans non imbriqués, $\mathbb{X}_n$ ($n \geqslant 2$) se construit à partir de tous les plans précédents : $\mathbb{X}_1$, $\cdots$, $\mathbb{X}_{n-1}$. Les matrices $\mathbb{X}_1$, $\cdots$, $\mathbb{X}_n$ sont notées :

$$\left\{\begin{array}{lcl}
\mathbb{X}_1 & = & \begin{pmatrix} x_{I_1}^{(1)} \\ \vdots \\ x_{I_1}^{(n_1)} \end{pmatrix}, \\
\mathbb{X}_2 & = & \begin{pmatrix} x_{I_1}^{(n_1+1)} & x_{I_2}^{(n_1+1)} \\ \vdots & \vdots \\ x_{I_1}^{(n_1+n_2)} & x_{I_2}^{(n_1+n_2)} \end{pmatrix}, \\
 & \vdots & \\
\mathbb{X}_{n-1} & = & \begin{pmatrix} x_{I_1}^{(n_1+\cdots+n_{n-2+1})} & \cdots & x_{I_{n-1}}^{(n_1+\cdots+n_{n-2+1})} \\ \vdots & \vdots & \vdots \\ x_{I_1}^{(n_1+\cdots+n_{n-2}+n_{n-1})} & \cdots & x_{I_{n-1}}^{(n_1+\cdots+n_{n-2}+n_{n-1})} \end{pmatrix}, \\
\mathbb{X}_n & = & \begin{pmatrix} x_{I_1}^{(n_1+\cdots+n_{n-1+1})} & \cdots & x_{I_{n-1}}^{(n_1+\cdots+n_{n-1+1})} & x_{I_n}^{(n_1+\cdots+n_{n-1+1})} \\ \vdots & \vdots & \vdots & \\ x_{I_1}^{(n_1+\cdots+n_{n-1}+n_n)} & \cdots & x_{I_{n-1}}^{(n_1+\cdots+n_{n-1}+n_n)} & x_{I_n}^{(n_1+\cdots+n_{n-1}+n_n)} \end{pmatrix}.
\end{array}\right.$$

Le plan $\mathbb{X}_n$ est construit par block de variables : d'abord les colonnes correspondant à x_{I_1} (notées $\mathbb{X}_{n,I_1}$), auxquelles sont ajoutées celles correspondant à x_{I_2} (notées $\mathbb{X}_{n,I_2}$) pour former $\mathbb{X}_{n,I_1\cup I_2}$, $\cdots$, enfin celles correspondant à x_{I_n} (notées $\mathbb{X}_{n,I_n}$), pour former $\mathbb{X}_n$ tout entier.

- Les colonnes $\mathbb{X}_{n,I_1}$ sont construites en générant un LHS dans $[0,1]^{d_1}$. Cet LHS est ensuite optimisé par recuit simulé pour minimiser le critère de maximin généralisé appliqué à la concaténation des colonnes correspondant à x_{I_1} des matrices $\mathbb{X}_1$ à $\mathbb{X}_n$: $\begin{pmatrix} x_{I_1}^{(1)} \\ \vdots \\ x_{I_1}^{(n_1+\cdots+n_n)} \end{pmatrix}$.
- Les colonnes $\mathbb{X}_{n,I_k}$ ($k \in [\![2, N]\!]$) sont construites en générant un plan $\mathbb{X}_{n,I_1\cup\cdots\cup I_k}$ (contenant les colonnes de $\mathbb{X}_n$ associées aux variables $(x_{I_1}, \cdots, x_{I_k})$) imbriqué dans $\mathbb{X}_{n,I_1\cup\cdots\cup I_{k-1}}$ (voir section précédente), minimisant la fonction objectif suivante :

$$\left(\sum_{\substack{x,t\in\overset{\sim}{\mathbb{X}}{}_k^n \\ x\neq t}} \left(\frac{1}{\|x-t\|^2_{[0,1]^{d_1+\cdots+d_n}}}\right)^{50}\right)^{\frac{1}{50}} + \left(\sum_{x\in\mathbb{X}_{n,I_1\cup\cdots\cup I_k}} \left(\frac{1}{\|x_{I_k}-\grave{x}_{I_k}\|^2_{[0,1]^{d_k}}}\right)^{50}\right)^{\frac{1}{50}},$$

avec :

$$\overset{\sim}{\mathbb{X}}{}_k^n = \begin{pmatrix} x_{I_1}^{(n_1+\cdots+n_{k-1}+1)} & \cdots & x_{I_k}^{(n_1+\cdots+n_{k-1}+1)} \\ \vdots & \vdots & \vdots \\ x_{I_1}^{(n_1+\cdots+n_{k-1}+\cdots+n_n)} & \cdots & x_{I_k}^{(n_1+\cdots+n_{k-1}+\cdots+n_n)} \end{pmatrix}.$$

8.3 Estimation des paramètres

La méthode choisie dans cette thèse est la maximisation de la vraisemblance. Ainsi, en notant $(\eta_1, \cdots, \eta_N)$ les paramètres respectifs de $(Y_1, \cdots, Y_N)$, la vraisemblance vaut :

$$\mathcal{L}(\eta_1, \cdots, \eta_N; \boldsymbol{y_1}, \cdots, \boldsymbol{y_N}) = h_Y(\boldsymbol{y}).$$

C'est la densité du vecteur $Y = (Y_1(\mathbb{X}_1), \cdots, Y_N(\mathbb{X}_N))$ appliquée aux observations $\boldsymbol{y} = (\boldsymbol{y_1}, \cdots, \boldsymbol{y_N})$. Dans le cas où les plans sont imbriqués, les observations de $(Z_n(\mathbb{X}_n))_{n=2}^N$ sont connues (et notées $(\boldsymbol{z_n})_{n=2}^N$). L'indépendance des Z_n conduit au découplage des vraisemblances :

$$\mathcal{L}(\eta_1, \cdots, \eta_N; \boldsymbol{y_1}, \cdots, \boldsymbol{y_N}) = \mathcal{L}(\eta_1; \boldsymbol{y_1}) \cdot \prod_{n=2}^{N} \mathcal{L}(\eta_n; \boldsymbol{z_n}).$$

Celles-ci peuvent alors être optimisées séparément ce qui simplifie l'estimation. Dans le cas où les plans ne sont pas imbriqués, la vraisemblance ne peut pas être découplée et un algorithme d'EM (Expectation-Maximisation) est utilisé. L'idée est de se ramener au cas découplé en considérant les $\overset{\sim}{\mathbb{X}}_n^N$, qui sont bien imbriqués les uns dans les autres, à la place des $\mathbb{X}_n$, et les observations de $Y_1(\overset{\sim}{\mathbb{X}}_1^N)$ et des $\left(Z_n(\overset{\sim}{\mathbb{X}}_n^N)\right)_{n=2}^N$ à la place de $\boldsymbol{y_1}$ et des $(\boldsymbol{z_n})_{n=2}^N$. La différence est que ces observations ne sont plus connues. Les vraisemblances $\mathcal{L}(\eta_n; \boldsymbol{z_n})$ (respectivement $\mathcal{L}(\eta_1; \boldsymbol{y_1})$) sont remplacées par leurs estimations a posteriori :

$$\begin{cases} \mathcal{Q}_1(\eta_1, \eta^*) &= \mathbb{E}_{\eta^*}\left[\mathcal{L}\left(\eta_1; Y_1\left(\overset{\sim}{\mathbb{X}}_1^N\right)\right) \mid Y = \boldsymbol{y}\right], \\ \mathcal{Q}_n(\eta_1, \eta^*) &= \mathbb{E}_{\eta^*}\left[\mathcal{L}\left(\eta_n; Z_n\left(\overset{\sim}{\mathbb{X}}_n^N\right)\right) \mid Y = \boldsymbol{y}\right]. \end{cases}$$

L'algorithme consiste alors a créer une suite $\left(\eta^{(i)}\right)_{i \geqslant 0}$ où, à chaque nouvelle itération $i+1$, les paramètres $\eta_n^{(i+1)}$ sont solutions de :

$$\max_{\eta_n} \mathcal{Q}_n(\eta_n, \eta_n^{(i)}).$$

L'estimation des paramètres, qui est censée être la limite de cette suite, est prise en tronquant la suite à partir d'un certain rang. L'algorithme EM permet de passer d'un seul problème d'optimisation complexe à plusieurs problèmes d'optimisation simples.

8.4 Cas test

Le métamodèle seqGPR est testé sur trois cas test : deux analytiques et un provenant d'un code de calcul industriel. Comme il y a beaucoup de paramètres à estimer dans ce métamodèle, une tentative de simplification est faite en imposant que des groupes de paramètres soient de même valeur. Ainsi chaque processus correctif Z_n $(n \geqslant 2)$ est doté d'un vecteur de paramètres de covariance de la forme :

$$\theta_n = \left(\underbrace{\alpha_n, \cdots, \alpha_n}_{I_1 \cup \cdots \cup I_{n-1}}, \underbrace{\theta_n^1, \cdots, \theta_n^{d_n}}_{I_n}\right).$$

La version de seqGPR avec ce regroupement de paramètres est appelée Robust, tandis que celle avec tous les paramètres dissociés est appelée Full. Sont également comparées les versions avec des processus correctifs de type Red ou de type P. Enfin, le métamodèle seqGPR est comparé à un krigeage classique entrainé sur tous les plans.

8.4.1 Cas test analytique en dimension 4

La fonction à approximer par un métamodèle est une fonction de quatre variables décomposée en la somme d'une fonction de deux variables et d'une fonction de quatre variables $f(x_1, x_2, x_3, x_4) = g_1(x_1, x_2) + g_2(x_1, x_2, x_3, x_4)$. Les deux fonctions sont définies de la manière suivante :

$$\begin{cases} g_1(x_1, x_2) & = & \left[4 - 2.1(4x_1 - 2)^2 + \frac{(4x_1-2)^4}{3}\right](4x_1 - 2)^2 \\ & + & (4x_1 - 2)(2x_2 - 1) + [-4 + 4(2x_2 - 1)^2](2x_2 - 1)^2, \\ g_2(x_1, x_2, x_3, x_4) & = & 4\exp\left(-\|x - 0.3\|^2\right). \end{cases}$$

g_1 est la fonction six-hump Camel redimensionnée dans $[0, 1]^2$. C'est une fonction assez chahutée. g_2 est une gaussienne beaucoup plus lisse. L'étude réalisée sur cette fonction est composée de deux étapes :

- A l'étape 1, les variables (x_1, x_2) sont libres et les variables (x_3, x_4) sont fixées aux valeurs $(\frac{x_1+x_2}{2}, 0.2x_1 + 0.7)$.
- A l'étape 2, toutes les variables sont libres.

Des plans de différentes natures sont générés durant ces deux étapes : imbriqués ou non-imbriqués, et de différentes tailles. Plusieurs résultats sont montrés par ce cas test. D'abord, les plans non-imbriqués donnent de meilleurs résultats à taille fixée. Ensuite, la version robuste du métamodèle seqGPR est meilleure que la version full et que le krigeage classique. Enfin, la version Red est plus performante que la version P. De ces conclusions sont tirées deux mesures appliquées dans la suite : l'utilisation de plans non-imbriqués et de la version robuste du métamodel seqGPR.

8.4.2 Cas test analytique en dimension 15

La nouvelle fonction à approximer est une fonction de 15 variables :

$$f \quad : \left| \begin{array}{lll} [-3, 3]^{15} & \to & \mathbb{R} \\ x & \mapsto & a_1' x + a_2' \sin x + a_3' \cos x + x' M x, \end{array} \right.$$

dont les entrées sont réarrangées par ordre décroissant d'indice de Sobol, et redimensionnées dans $[0, 1]^{15}$. Cette fois-ci, une étude de trois étapes est réalisée : libérant les trois premières variables, puis les 6 suivantes, et enfin les 6 dernières. Des plans d'entraînement $\mathbb{X}_1$, $\mathbb{X}_2$, $\mathbb{X}_3$ sont générés, respectivement de taille 15, 45, et 75. A chaque étape, les variables fixées sont prises de valeur 0.5. Les mêmes conclusions sont tirées par rapport au cas test précédent : à savoir que le métamodèle seqGPR est meilleur que le krigeage classique, et que la version Red est meilleure que la version P.

8.4.3 Cas test industriel en dimension 15

Le dernier cas test, qui est celui ayant motivé cette thèse, comporte également 15 entrées. Une étude de deux étapes est réalisée : dans la première, les 11 premières variables sont libres et les 4 dernières sont fixées à une constante, dans la deuxième, toutes les variables sont libres. Un plan est réalisé à l'étape 1, de taille 126, et un autre à l'étape 2, de taille 299. L'évaluation des métamodèles est réalisée sous la forme d'une validation croisée. $\mathbb{X}_1$ prend 50 points du plan généré à l'étape 1, tandis que $\mathbb{X}_2$ prend 50 points parmi ceux générés à l'étape

2, laissant les autres comme ensemble test. 30 évaluations des métamodèles sont réalisées (avec des $\mathbb{X}_1$ et $\mathbb{X}_2$ différents à chaque fois). Encore une fois, le métamodèle seqGPR a de meilleurs résultats de prédiction que le krigeage classique, cette fois-ci avec un léger avantage pour la version P.

8.5 Tentatives d'approfondissement de la méthode

8.5.1 Conditionnement multiple

Dans cette sous-section, les études changent légèrement de forme. A chaque étape $n \in [\![1, N-1]\!]$, plusieurs plans sont générés dans différents sous-espaces : $\mathbb{X}_n^1, \cdots, \mathbb{X}_n^{K_{n+1}}$. Ces K_{n+1} sous-espaces sont liés par le fait que les variables $x_{I_{n+2}}, \cdots, x_{I_N}$ sont fixées aux mêmes valeurs : $\grave{x}_{I_{n+2}}, \cdots, \grave{x}_{I_N}$ (qui dépendent des variables $x_{I_1}, \cdots, x_{I_{n+1}}$), et que d'un autre côté, les valeurs des variables $x_{I_{n+1}}$ dans ces sous-espaces diffèrent d'une constante b_{n+1}^i, valant $\grave{x}_{I_{n+1}} + b_{n+1}^i$ dans le sous-espace $i \in [\![1, K_{n+1}]\!]$. La dernière étape N est inchangée, où un unique plan $\mathbb{X}_N$ est généré dans l'espace tout entier. Cette multiplicité des sous-espaces à étape fixée oblige à une modification de la modélisation dans laquelle les processus des étapes intermédiaires doivent prendre en compte un groupe de variable supplémentaire:

$$\left\{\begin{array}{lcll} Y_1(x_{I_1}, x_{I_2}) & = & m + Z_1(x_{I_1}, x_{I_2}), & \\ Y_n(x_{I_1\cup\cdots\cup I_n}, x_{I_{n+1}}) & = & Y_{n-1}(x_{I_1\cup\cdots\cup I_n}) + Z_n(x_{I_1\cup\cdots\cup I_n}, x_{I_{n+1}}), & \forall n \in [\![2, N-1]\!], \\ Y_N(x_{I_1}, \cdots, x_{I_N}) & = & Y_{N-1}(x_{I_1}, \cdots, x_{I_N}) + Z_N(x_{I_1}, \cdots, x_{I_N}). & \end{array}\right.$$

La nouvelle propriété de nullité vérifiée par les $(Z_n)_{n=2}^N$ est :

$$\left\{\begin{array}{lcll} Z_n(x_{I_1\cup\cdots I_{n-1}}, \grave{x}_{I_n} + \mathbb{B}_n, \grave{x}_{I_{n+1}}) & = & \mathbf{0}, & \forall n \in [\![2, N-1]\!], \forall x_{I_1\cup\cdots I_{n-1}} \in [0,1]^{d_1+\cdots+d_{n-1}}, \\ Z_N(x_{I_1\cup\cdots\cup I_{N-1}}, \grave{x}_{I_N} + \mathbb{B}_N) & = & \mathbf{0}, & \forall x_{I_1\cup\cdots\cup I_{N-1}} \in [0,1]^{d_1+\cdots+d_{N-1}}. \end{array}\right.$$

avec $\mathbb{B}_n = \begin{pmatrix} b_n^1 \\ \vdots \\ b_n^{K_n} \end{pmatrix}$, pour tout $n \geqslant 2$.

Si la méthodologie d'estimation par EM fonctionne encore dans ce nouveau format, il faut reconstruire les $(Z_n)_{n=2}^N$ dont la propriété de nullité est désormais multiple. Le choix est porté sur une généralisation du processus P à plusieurs plans :

$$\left\{\begin{array}{l} Z_n(x_{I_1\cup\cdots I_{n-1}}, x_{I_n}, x_{I_{n+1}}) = \widetilde{Z}_n(x_{I_1\cup\cdots I_{n-1}}, x_{I_n}, x_{I_{n+1}}) \\ -\mathbb{E}\left[\widetilde{Z}_n(x_{I_1\cup\cdots I_{n-1}}, x_{I_n}, x_{I_{n+1}}) \mid \widetilde{Z}_n(\mathcal{D}_n)\right], \qquad \forall n \in [\![2, N-1]\!], \\ \\ Z_N(x_{I_1\cup\cdots I_{N-1}}, x_{I_N}) = \widetilde{Z}_N(x_{I_1\cup\cdots I_{N-1}}, x_{I_N}) \\ -\mathbb{E}\left[\widetilde{Z}_N(x_{I_1\cup\cdots I_{N-1}}, x_{I_N}) \mid \widetilde{Z}_N(\mathcal{D}_N)\right]. \end{array}\right.$$

où $\mathcal{D}_n = \{(t_{I_1\cup\cdots I_{n-1}}, \grave{t}_{I_n} + \mathbb{B}_n, \grave{t}_{I_{n+1}}), t_{I_1\cup\cdots I_{n-1}} \in [0,1]^{d_1+\cdots+d_{n-1}}\}$ est la zone de nullité à l'étape n et $\mathcal{D}_N = \{(t_{I_1\cup\cdots I_{N-1}}, \grave{t}_{I_N} + \mathbb{B}_N), t_{I_1\cup\cdots I_{N-1}} \in [0,1]^{d_1+\cdots+d_{N-1}}\}$ est la zone de nullité à l'étape N. Une nouvelle fois, une forme particulière des noyaux des processus latents $\left(\widetilde{Z}_n\right)_{n=2}^N$ qui est la même que dans le cas présenté dans la section 8.1 permet de donner une forme explicite

au processus P :

$$\begin{cases} k_n((x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n\cup I_{n+1}}), (t_{I_1\cup\cdots\cup I_{n-1}}, t_{I_n\cup I_{n+1}})) = \\ \sigma_n^2 r_n^1(x_{I_1\cup\cdots\cup I_{n-1}}, t_{I_1\cup\cdots\cup I_{n-1}}) r_n^2(x_{I_n\cup I_{n+1}} - \dot{x}_{I_n\cup I_{n+1}}, t_{I_n\cup I_{n+1}} - \dot{t}_{I_n\cup I_{n+1}}), \quad \forall n \in [\![2, N-1]\!], \\ \\ k_N((x_{I_1\cup\cdots\cup I_{N-1}}, x_{I_N}), (t_{I_1\cup\cdots\cup I_{N-1}}, t_{I_N})) = \\ \sigma_N^2 r_N^1(x_{I_1\cup\cdots\cup I_{N-1}}, t_{I_1\cup\cdots\cup I_{N-1}}) r_N^2(x_{I_N} - \dot{x}_{I_N}, t_{I_N} - \dot{t}_{I_N}). \end{cases}$$

Dans ce cas les processus P valent :

$$\begin{cases} Z_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n\cup I_{n+1}}) = \widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, x_{I_n\cup I_{n+1}}) \\ -r_n^2(x_{I_n\cup I_{n+1}}, \mathcal{B}_n) r_n^2(\mathcal{B}_n, \mathcal{B}_n) \widetilde{Z}_n(x_{I_1\cup\cdots\cup I_{n-1}}, \dot{x}_{I_n\cup I_{n+1}} + \mathcal{B}_n), \quad \forall n \in [\![2, N-1]\!], \\ \\ Z_N(x_{I_1\cup\cdots\cup I_{N-1}}, x_{I_N}) = \widetilde{Z}_N(x_{I_1\cup\cdots\cup I_{N-1}}, x_{I_N}) \\ -r_N^2(x_{I_N}, \mathbb{B}_N) r_N^2(\mathbb{B}_N, \mathbb{B}_N) \widetilde{Z}_N(x_{I_1\cup\cdots\cup I_{N-1}}, \dot{x}_{I_N} + \mathbb{B}_N), \end{cases}$$

avec $\mathcal{B}_n = \begin{pmatrix} b_n^1 & 0 \\ \vdots & \vdots \\ b_n^{K_n} & 0 \end{pmatrix}$, de sorte que $\dot{x}_{I_n\cup I_{n+1}} + \mathcal{B}_n = \left(\dot{x}_{I_n} + \mathbb{B}_n, \dot{x}_{I_{n+1}}\right)$.

L'évaluation de cette adaptation du métamodèle seqGPR sur les cas test précédents ne révèle aucune amélioration par rapport au krigeage classique dans ce cas. Sans doute la complexité trop élevée du modèle n'est cette fois-ci pas compensée par son adaptation au format des données d'entraînement.

8.5.2 Enrichissement des plans

La deuxième piste d'approfondissement est l'enrichissement des plans d'expérience, dans le cadre de plans non-imbriqués. L'enrichissement se fait en sélectionnant un point solution d'un certain problème d'optimisation. Le choix se porte sur le problème d'optimisation suivant :

$$\begin{cases} \max\limits_{x\in\mathcal{X}} \hat{v}(x) \left[1 + \sum\limits_{i=1}^{n} \dfrac{(y_i - \hat{y}_{-i}(x^{(i)}))^2}{\hat{v}_{-i}(x^{(i)})} \mathbf{1}_{x\in V_i}\right], \\ \\ \text{sous les contraintes :} \quad \left| \begin{array}{l} dist\left(x_{I_1\cup\cdots\cup I_n}, \overset{\sim}{\mathbb{X}}_n^N\right) > u_n, \ \forall n \in [\![1, N]\!] \\ dist(x_{I_n} - \dot{x}_{I_n}) > v_n, \ \forall n \in [\![2, N]\!] \end{array} \right. \end{cases}$$

V_i designe la cellule de Voronoï du point d'entraînement $x^{(i)}$. La fonction objectif veille à ce que le point soit choisi dans une zone de forte variance de prédiction $\hat{v}(x)$, c'est-à-dire loin des points d'entraînement déjà présents, mais également dans une zone où l'erreur de prédiction réelle est potentiellement forte (ici c'est l'erreur de prédiction du point d'entraînement le plus proche qui donne une idée de sa valeur). Les contraintes veillent à ce que les $\overset{\sim}{\mathbb{X}}_n^N$ aient des points suffisamment éloignés les uns des autres, et que le point x soit éloigné de tous les sous-espaces des étapes précédentes, le but ultime étant que les matrices mises en jeu dans l'EM soient inversibles.

Etant donné que l'enrichissement ne se fait pas point par point mais plutôt par paquets de points choisis en même temps, une procédure gloutonne utilisant le problème d'optimisation

précédemment défini est mise en place. Le paquet de points, dont la taille est imposée par l'utilisateur, est sélectionné parmi une suite de Sobol qui forme un ensemble de points candidats. La procédure consiste à, de manière séquentielle, retirer les points candidats ne vérifiant pas les contraintes, puis choisir le point candidat optimal pour la fonction objectif, ensuite mettre à jour le critère de prédiction avec ce nouveau point, et enfin recommencer jusqu'à avoir atteint la taille requise. Le paquet ainsi confectionné peut alors enfin être soumis au code de calcul, pour en connaître la valeur de sortie, et ajouté à l'ensemble d'entraînement du métamodèle seqGPR.

La comparaison de cette méthode d'enrichissement avec d'autres méthodes comme le maximum de variance de prédiction ou encore la sélection d'un paquet aléatoire donne des résultats assez différents suivant les cas test. Il semble que globalement cette méthode d'enrichissement ne soit pas très performante tant que le métamodèle n'est pas assez prédictif.

Chapter 9

Appendix

9.1 Example in 4D

This section develops the example in 4D, introduced in subsection 3.1.2 in chapter 3, for the different chapters of the thesis.

9.1.1 Illustration of chapter 3

Red process

- If Z_2 is a Red process, it is defined as

$$Z_2(x_1, x_2, x_3) = \widetilde{Z}_2(x_1, x_2, x_3) - \widetilde{Z}_2(x_1, 0.4, 0.5).$$

  Its covariance kernel is equal to $\sigma_2^2 \rho_2$ with

$$\begin{array}{rcl} \rho_2\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right) & = & r_2\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right) \\ & + & r_2\left((x_1, 0.4, 0.5), (t_1, 0.4, 0.5)\right) \\ & - & r_2\left((x_1, x_2, x_3), (t_1, 0.4, 0.5)\right) \\ & - & r_2\left((x_1, 0.4, 0.5), (t_{1,2}, t_3)\right). \end{array}$$

- If Z_3 is a Red process, it is defined as

$$Z_3(x_1, x_2, x_3, x_4) = \widetilde{Z}_3(x_1, x_2, x_3, x_4) - \widetilde{Z}_3\left(x_1, x_2, x_3, \frac{x_1 + x_2 + x_3}{4}\right).$$

  Its covariance kernel is equal to $\sigma_3^2 \rho_3$ with

$$\begin{array}{rcl} \rho_3\left((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4)\right) & = & r_3\left((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4)\right) \\ & + & r_3\left((x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}), (t_1, t_2, t_3, \frac{t_1+t_2,t_3}{4})\right) \\ & - & r_3\left((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, \frac{t_1+t_2+t_3}{4})\right) \\ & - & r_3\left((x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}), (t_1, t_2, t_3, t_4)\right) \end{array}$$

Psi process

- If Z_2 is a Psi process, it is defined as

$$Z_2(x_1, x_2, x_3) = \Psi_2\left(x_2 - 0.4, x_3 - 0.5\right) \widetilde{Z}_2(x_1, x_2, x_3),$$

with for all (t_1, t_2) in $[-1, 1]^2$, $\Psi_2(t_1, t_2) = 1 - \exp\left(-\frac{t_1^2}{2\delta_{2,1}^2} - \frac{t_2^2}{2\delta_{2,2}^2}\right)$. Its covariance kernel is equal to $\sigma_2^2\rho_2$ with

$$\begin{aligned}\rho_2\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right) &= \Psi_2\left(x_2 - 0.4, x_3 - 0.5\right) \\ &\times \Psi_2\left(t_2 - 0.4, t_3 - 0.5\right) \\ &\times r_2\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right).\end{aligned}$$

- If Z_3 is a Psi process, it is defined as

$$Z_3(x_1, x_2, x_3, x_4) = \Psi_3\left(x_4 - \frac{x_1 + x_2 + x_3}{4}\right) \widetilde{Z}_3(x_1, x_2, x_3, x_4),$$

with for all t in $[0, 1]$, $\Psi_3(t) = 1 - \exp\left(-\frac{t^2}{2\delta_3^2}\right)$. Its covariance kernel is equal to $\sigma_3^2\rho_3$ with

$$\begin{aligned}\rho_3\left((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4)\right) &= \Psi_3\left(x_4 - \tfrac{x_1+x_2+x_3}{4}\right) \\ &\times \Psi_3\left(t_4 - \tfrac{t_1+t_2+t_3}{4}\right) \\ &\times r_3\left((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4)\right).\end{aligned}$$

P process

- If Z_2 is a P process, it is defined as

$$Z_2(x_1, x_2, x_3) = \widetilde{Z}_2(x_1, x_2, x_3) - r_{2,3}\left((x_2, x_3), (0.4, 0.5)\right) \widetilde{Z}_2(x_1, 0.4, 0.5).$$

where $\widetilde{Z}_2 \sim \mathcal{GP}\left(0, \sigma_2^2 r_1\left(x_1, t_1\right) r_{2,3}\left((x_1, x_2), (t_1, t_2)\right)\right)$ with r_1 and $r_{2,3}$ stationary correlation kernels. Its covariance kernel is equal to $\sigma_2^2\rho_2$ with

$$\begin{aligned}\rho_2\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right) &= r_1\left(x_1, t_1\right) \times \left[r_{2,3}\left((x_2, x_3), (t_2, t_3)\right)\right. \\ &- \left. r_{2,3}\left((x_2, x_3), (0.4, 0.5)\right) r_{2,3}\left((0.4, 0.5), (t_2, t_3)\right)\right].\end{aligned}$$

- If Z_3 is a P process, it is defined as

$$\begin{aligned}Z_3(x_1, x_2, x_3, x_4) &= \widetilde{Z}_3(x_1, x_2, x_3, x_4) \\ &- r_4\left(x_4 - \tfrac{x_1+x_2+x_3}{4}, 0\right) \widetilde{Z}_3\left(x_1, x_2, x_3, \tfrac{x_1+x_2+x_3}{4}\right).\end{aligned}$$

Where $\widetilde{Z}_3 \sim \mathcal{GP}\left(0, \sigma_3^2 r_{1,2,3}\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right) r_4\left(x_4 - \frac{x_1+x_2+x_3}{4}, t_4 - \frac{t_1+t_2+t_3}{4}\right)\right)$ with $r_{1,2,3}$ and r_4 stationary correlation kernels. Its covariance kernel is equal to $\sigma_3^2\rho_3$ with

$$\begin{aligned}\rho_3\left((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4)\right) &= r_{1,2,3}\left((x_1, x_2, x_3), (t_1, t_2, t_3)\right) \\ &\times \left[r_4\left(x_4 - \tfrac{x_1+x_2+x_3}{4}, t_4 - \tfrac{t_1+t_2+t_3}{4}\right)\right. \\ &- \left. r_4\left(x_4 - \tfrac{x_1+x_2+x_3}{4}, 0\right) r_4\left(0, t_4 - \tfrac{t_1+t_2+t_3}{4}\right)\right].\end{aligned}$$

9.1.2 Illustration of chapter 4

Nested designs The designs $\mathbb{X}_1$, $\mathbb{X}_2$, and $\mathbb{X}_3$ are nested ($\mathbb{X}_3 \sqsubset \mathbb{X}_2 \sqsubset \mathbb{X}_1$) if they are of the form:

$$\begin{cases} \mathbb{X}_1 &= \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}, \\ \mathbb{X}_2 &= \begin{pmatrix} x_1^{(i_1)} & x_2^{(i_1)} & x_3^{(i_1)} \\ \vdots & \vdots & \vdots \\ x_1^{(i_{n_2})} & x_2^{(i_{n_2})} & x_3^{(i_{n_2})} \end{pmatrix}, & \text{with } \{i_1, \cdots, i_{n_2}\} \subset \{1, \cdots, n_1\}, \\ \mathbb{X}_3 &= \begin{pmatrix} x_1^{(j_1)} & x_2^{(j_1)} & x_3^{(j_1)} & x_4^{(j_1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(j_{n_3})} & x_2^{(j_{n_3})} & x_3^{(j_{n_3})} & x_4^{(j_{n_3})} \end{pmatrix}, & \text{with } \{j_1, \cdots, j_{n_3}\} \subset \{i_1, \cdots, i_{n_2}\}. \end{cases}$$

In order to compute the likelihoods and the predictions, the matrices $cov(Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_1)) = \sigma_1^2 \rho_{\theta_1}(\mathbb{X}_1, \mathbb{X}_1)$, $cov(Z_2(\mathbb{X}_2), Z_2(\mathbb{X}_2)) = \sigma_2^2 \rho_{\theta_2}(\mathbb{X}_2, \mathbb{X}_2)$, and $cov(Z_3(\mathbb{X}_3), Z_3(\mathbb{X}_3)) = \sigma_3^2 \rho_{\theta_3}(\mathbb{X}_3, \mathbb{X}_3)$ must be invertible. $\mathbb{X}_1$ must be composed of distinct points sufficiently far from each other to ensure the invertibility of the first matrix. Because of the nullity property of Z_2: $Z_2(x_1, 0.4, 0.5) = 0$ for all x_1 in $[0, 1]$ (resp. Z_3: $Z_3(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}) = 0$ for all $(x_1, x_2, x_3) \in [0, 1]^3$), its covariance kernel is not positive definite, therefore the same constraint on $\mathbb{X}_2$ (resp. $\mathbb{X}_3$) is not sufficient. An additional constraint must be added, that the points of $\mathbb{X}_2$ (resp. $\mathbb{X}_3$) are sufficiently far from the nullity subspace $\mathcal{D}_2 = \{(x_1, 0.4, 0.5), x_1 \in [0, 1]\}$ (resp. $\mathcal{D}_3 = \{(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}), (x_1, x_2, x_3) \in [0, 1]^3\}$), i.e. for all k in $[\![1, n_2]\!]$, $(x_2^{(i_k)}, x_3^{(i_k)})$ must be sufficiently far from $(0.4, 0.5)$ (resp. for all k in $[\![1, n_3]\!]$, $x_4^{(j_k)}$ must be sufficiently far from $\frac{x_1^{(j_k)}+x_2^{(j_k)}+x_3^{(j_k)}}{4}$). This second constraint ensures that the covariance matrix has no rows full of zeros.

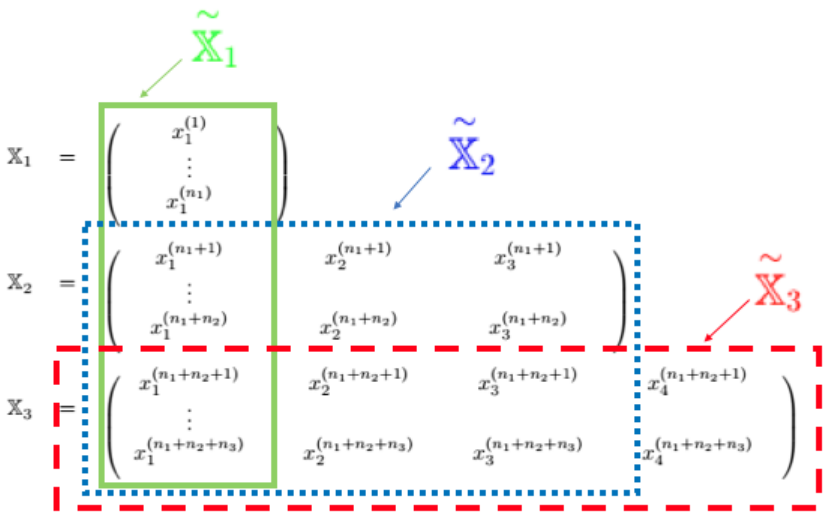
Non-nested designs The designs $\mathbb{X}_1$, $\mathbb{X}_2$ and $\mathbb{X}_3$ are non-nested if they are of the form:

$$\begin{cases} \mathbb{X}_1 &= \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}, \\ \mathbb{X}_2 &= \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} & x_3^{(n_1+n_2)} \end{pmatrix}, \\ \mathbb{X}_3 &= \begin{pmatrix} x_1^{(n_1+n_2+1)} & x_2^{(n_1+n_2+1)} & x_3^{(n_1+n_2+1)} & x_4^{(n_1+n_2+1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2+n_3)} & x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} & x_4^{(n_1+n_2+n_3)} \end{pmatrix}, \end{cases}$$

with the following constraints:

$$
\begin{cases}
\begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix} \cap \begin{pmatrix} x_1^{(n_1+1)} \\ \vdots \\ x_1^{(n_1+n_2)} \end{pmatrix} = \emptyset, \\
\begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1+n_2)} \end{pmatrix} \cap \begin{pmatrix} x_1^{(n_1+n_2+1)} \\ \vdots \\ x_1^{(n_1+n_2+n_3)} \end{pmatrix} = \emptyset, \\
\begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} & x_3^{(n_1+n_2)} \end{pmatrix} \cap \begin{pmatrix} x_1^{(n_1+n_2+1)} & x_2^{(n_1+n_2+1)} & x_3^{(n_1+n_2+1)} \\ \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2+n_3)} & x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} \end{pmatrix} = \emptyset.
\end{cases}
$$

An illustration of the different $\widetilde{\mathbb{X}}_n$ is shown in figure below.



To compute $\mathcal{Q}_1$, $\mathcal{Q}_2$ and $\mathcal{Q}_3$, the designs $\widetilde{\mathbb{X}}_1$, $\widetilde{\mathbb{X}}_2$, and $\widetilde{\mathbb{X}}_3$ must have distinct points sufficiently far from each other. $\widetilde{\mathbb{X}}_2$ (respectively $\widetilde{\mathbb{X}}_3$) should have points sufficiently far from the nullity subset $\mathcal{D}_2 = \{(x_1, 0.4, 0.5), x_1 \in [0,1]\}$ (resp. $\mathcal{D}_3 = \{(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}), (x_1, x_2, x_3) \in [0,1]^3\}$). For all k in $[\![1, n_2 + n_3]\!]$, $(x_2^{(n_1+k)}, x_3^{(n_1+k)})$ must be far from $(0.4, 0.5)$, and for all k in $[\![1, n_3]\!]$, $x_4^{(n_1+n_2+k)}$ must be far from $\frac{x_1^{(n_1+n_2+k)}+x_2^{(n_1+n_2+k)}+x_3^{(n_1+n_2+k)}}{4}$.

9.1.3 Illustration of chapter 5

Nested designs

- The Φ_2 criterion applied to $\mathbb{X}_2$ is equal to:

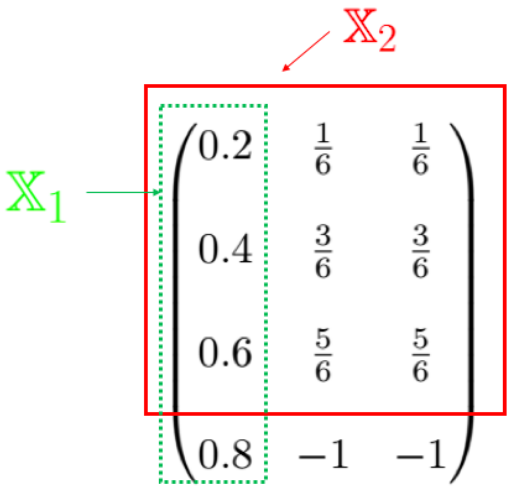
$$
\begin{aligned}
\Phi_2(\mathbb{X}_2) &= \left(\sum_{k=1}^{n_2} \sum_{\substack{l=1 \\ l \neq k}}^{n_2} \left(\frac{1}{\left\| x^{(i_k)} - x^{(i_l)} \right\|_{[0,1]^3}^2} \right)^{50} \right)^{\frac{1}{50}} \\
&+ \left(\sum_{k=1}^{n_2} \left(\frac{1}{(x_2^{(i_k)} - 0.4)^2 + (x_3^{(i_k)} - 0.5)^2} \right)^{50} \right)^{\frac{1}{50}}.
\end{aligned}
$$

- The Φ_3 criterion applied to $\mathbb{X}_3$ is equal to:

$$\Phi_3(\mathbb{X}_3) = \left(\sum_{k=1}^{n_3}\sum_{\substack{l=1\\l\neq k}}^{n_3}\left(\frac{1}{\left\|x^{(j_k)}-x^{(j_l)}\right\|^2_{[0,1]^4}}\right)^{50}\right)^{\frac{1}{50}} + \left(\sum_{k=1}^{n_3}\left(\frac{1}{(x_4^{(j_k)}-\frac{x_1^{(j_k)}+x_2^{(j_k)}+x_3^{(j_k)}}{4})^2}\right)^{50}\right)^{\frac{1}{50}}.$$

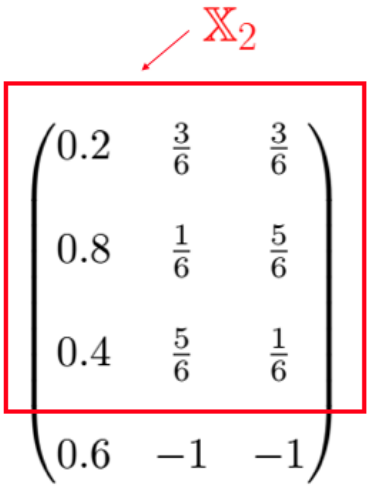
The building algorithm is illustrated below.

- At step 1, a design $\mathbb{X}_1 = \begin{pmatrix}0.2\\0.4\\0.6\\0.8\end{pmatrix}$ has been generated. The values of x_2 and x_3 are respectively fixed to 0.4 and 0.5. The values of x_4 are fixed to $\frac{x_1+x_2+x_3}{4}$.
- At step 2, the goal is to generate a design $\mathbb{X}_2$ of size 3 nested in $\mathbb{X}_1$. The following matrix is defined $\begin{pmatrix}0.2 & \frac{1}{6} & \frac{1}{6}\\0.4 & \frac{3}{6} & \frac{3}{6}\\0.6 & \frac{5}{6} & \frac{5}{6}\\0.8 & -1 & -1\end{pmatrix}$, from which the initialization of $\mathbb{X}_2$ is deduced by keeping the first three rows. The initialization of the design $\mathbb{X}_2$ is illustrated in figure below.



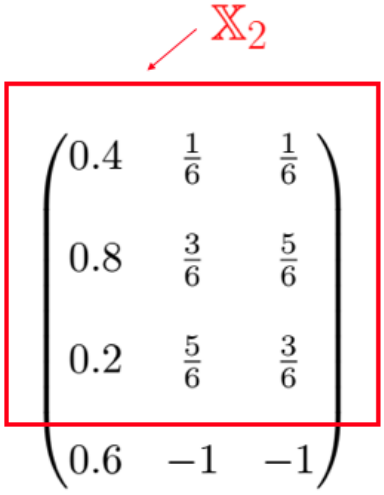
$\Phi_2(\mathbb{X}_2) = 11.98.$

An example of candidate for $\mathbb{X}_2$ resulting from permutations of the rows of the initial candidate is illustrated in figure below. It is obtained by performing the permutation $\begin{pmatrix}1 & 4 & 2 & 3\end{pmatrix}$ in the first column, the permutation $\begin{pmatrix}2 & 1 & 3\end{pmatrix}$ in the second column, and the permutation $\begin{pmatrix}2 & 3 & 1\end{pmatrix}$ in the third column.



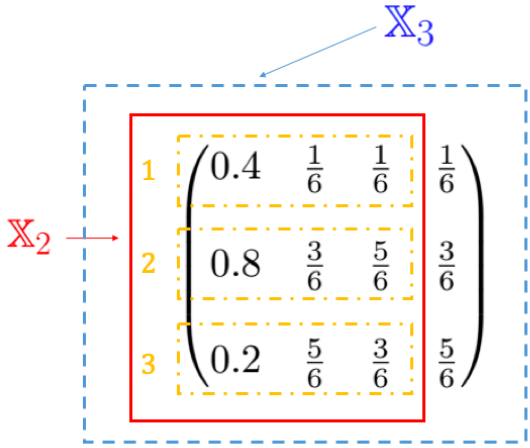
$\Phi_2(\mathbb{X}_2) = 11.95.$

The optimal candidate for the criteria Φ_2 is shown in figure below. It is found using algorithm 1 in appendix 9.4.2.



$\Phi_2(\mathbb{X}_2) = 4.20.$

- At step 3, the goal is to generate a design $\mathbb{X}_3$ of size 3 nested in $\mathbb{X}_2$. The initialization of the design $\mathbb{X}_3$ is illustrated below:



$\Phi_3(\mathbb{X}_3) = 61.20.$

The possible candidates for $\mathbb{X}_3$ result from permutations of rows in the initial candidate. Two kinds of permutations of rows are possible, by block (the blocks are represented by the orange rectangles with mixed dashes) in the first three columns, by component in the 4th column. An example of candidate is shown in figure below. It is built by applying the permutation $\begin{pmatrix}2 & 1 & 3\end{pmatrix}$ for the blocks of the first three columns, and the permutation $\begin{pmatrix}1 & 3 & 2\end{pmatrix}$ in the last column.

$$\begin{matrix}2\\1\\3\end{matrix}\begin{pmatrix}0.8 & \frac{3}{6} & \frac{5}{6} & \frac{1}{6}\\ 0.4 & \frac{1}{6} & \frac{1}{6} & \frac{5}{6}\\ 0.2 & \frac{5}{6} & \frac{3}{6} & \frac{3}{6}\end{pmatrix} \qquad \Phi_3(\mathbb{X}_3) = 9.78.$$

The optimal candidate for the criterion Φ_3 is shown below. It is found using algorithm 1 in appendix 9.4.2.

$$\begin{matrix}3\\2\\1\end{matrix}\begin{pmatrix}0.2 & \frac{5}{6} & \frac{3}{6} & \frac{5}{6}\\ 0.8 & \frac{3}{6} & \frac{5}{6} & \frac{1}{6}\\ 0.4 & \frac{1}{6} & \frac{1}{6} & \frac{3}{6}\end{pmatrix} \qquad \Phi_3(\mathbb{X}_3) = 4.35.$$

Non-nested designs

- $\mathbb{X}_1$ is of the form $\mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}$.
- $\mathbb{X}_2$ is of the form $\mathbb{X}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} & x_3^{(n_1+n_2)} \end{pmatrix}$. It must ensure that the design:

$$\overset{\sim}{\mathbb{X}}{}_1^2 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \\ x_1^{(n_1+1)} \\ \vdots \\ x_1^{(n_1+n_2)} \end{pmatrix},$$

is space-filling with points distant from each other in $[0,1]$ (it is the part of $\overset{\sim}{\mathbb{X}}_1$ concerned at the time, here $\mathbb{X}_3$ is not taken into account). $\mathbb{X}_2$ itself (equal to $\overset{\sim}{\mathbb{X}}_2^2$) must be space-filling with remote points in $[0,1]^3$ and verifying that $(x_2^{(n_1+i)}, x_3^{(n_1+i)})$ $(i \in [\![1, n_2]\!])$ is far from $(0.4, 0.5)$. $\mathbb{X}_2$ is built in two phases:

- The first column $\mathbb{X}_{2,I_1} = \begin{pmatrix} x_1^{(n_1+1)} \\ \vdots \\ x_1^{(n_1+n_2)} \end{pmatrix}$ ($I_1 = \{1\}$ is the index set of the variables released at step 1) is built first to ensure the properties of $\overset{\sim}{\mathbb{X}}_1^2$. $\mathbb{X}_{2,I_1}$ must minimize:

$$\Phi_{2,1}(\mathbb{X}_{2,I_1}) = \left(\sum_{i=1}^{n_1+n_2} \sum_{\substack{j=1 \\ j \neq i}}^{n_1+n_2} \left(\frac{1}{(x_1^{(i)} - x_1^{(j)})^2} \right)^{50} \right)^{\frac{1}{50}}$$

  which is the generalized maximin criterion applied to $\overset{\sim}{\mathbb{X}}_1^2$.

- The columns two and three $\mathbb{X}_{2,I_2} = \begin{pmatrix} x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ \vdots & \vdots \\ x_2^{(n_1+n_2)} & x_3^{(n_1+n_2)} \end{pmatrix}$ ($I_2 = \{2,3\}$ is the index set of the variables released at step 2) are built together to ensure the properties of $\overset{\sim}{\mathbb{X}}_2^2$. The goal is to create the design $\mathbb{X}_2$ such that it is nested in $\mathbb{X}_{2,I_1}$. $\mathbb{X}_{2,I_2}$ must minimize:

$$\begin{aligned} \Phi_{2,2}(\mathbb{X}_{2,I_2}) &= \left(\sum_{\substack{i,j=n_1+1 \\ j \neq i}}^{n_1+n_2} \left(\frac{1}{(x_1^{(i)}-x_1^{(j)})^2+(x_2^{(i)}-x_2^{(j)})^2+(x_3^{(i)}-x_3^{(j)})^2} \right)^{50} \right)^{\frac{1}{50}} \\ &+ \left(\sum_{i=n_1+1}^{n_1+n_2} \left(\frac{1}{(x_2^{(i)}-0.4)^2+(x_3^{(i)}-0.5)^2} \right)^{50} \right)^{\frac{1}{50}}. \end{aligned}$$

  The first part is the generalized maximin applied to $\overset{\sim}{\mathbb{X}}_2^2$, the second part ensures that $\mathbb{X}_2$ is far from the subspace where Z_2 is null.

- $\mathbb{X}_3$ is of the form $\mathbb{X}_3 = \begin{pmatrix} x_1^{(n_1+n_2+1)} & x_2^{(n_1+n_2+1)} & x_3^{(n_1+n_2+1)} & x_4^{(n_1+n_2+1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2+n_3)} & x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} & x_4^{(n_1+n_2+n_3)} \end{pmatrix}$. It must ensure that $\overset{\sim}{\mathbb{X}}_1^3 = \overset{\sim}{\mathbb{X}}_1$ is space-filling with points remote from each other in $[0,1]$. It must also ensure that $\overset{\sim}{\mathbb{X}}_2^3 = \overset{\sim}{\mathbb{X}}_2$ is space-filling with remote points in $[0,1]^3$ and verifying that $(x_2^{(n_1+i)}, x_3^{(n_1+i)})$ $(i \in [\![1, n_2+n_3]\!])$ is far from $(0.4, 0.5)$. $\mathbb{X}_3$ itself (equal to $\overset{\sim}{\mathbb{X}}_3^3$) must be space-filling with remote points in $[0,1]^4$ and verifying that $x_4^{(n_1+n_2+i)}$ $(i \in [\![1, n_3]\!])$ is far from $\frac{x_1^{(n_1+n_2+i)}+x_2^{(n_1+n_2+i)}+x_3^{(n_1+n_2+i)}}{4}$. $\mathbb{X}_3$ is built in three phases:

  - The first column $\mathbb{X}_{3,I_1} = \begin{pmatrix} x_1^{(n_1+n_2+1)} \\ \vdots \\ x_1^{(n_1+n_2+n_3)} \end{pmatrix}$ is built to ensure the properties of $\overset{\sim}{\mathbb{X}}_1$.

$\mathbb{X}_{3,I_1}$ must minimize:

$$\Phi_{3,1}(\mathbb{X}_{3,I_1}) = \left(\sum_{i=1}^{n_1+n_2+n_3} \sum_{\substack{j=1 \\ j\neq i}}^{n_1+n_2+n_3} \left(\frac{1}{(x_1^{(i)} - x_1^{(j)})^2} \right)^{50} \right)^{\frac{1}{50}}$$

which is the generalized maximin criterion applied to $\widetilde{\mathbb{X}}_1$.

– The columns two and three $\mathbb{X}_{3,I_2} = \begin{pmatrix} x_2^{(n_1+n_2+1)} & x_3^{(n_1+n_2+1)} \\ \vdots & \vdots \\ x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} \end{pmatrix}$ are built together to ensure the properties of $\widetilde{\mathbb{X}}_2$. The goal is to create the design $\mathbb{X}_{3,I_1\cup I_2}$ grouping the first three columns of $\mathbb{X}_3$ such that it is nested in $\mathbb{X}_{3,I_1}$. $\mathbb{X}_{3,I_2}$ must minimize:

$$\begin{aligned} \Phi_{3,2}(\mathbb{X}_{3,I_2}) &= \left(\sum_{\substack{i,j=n_1+1 \\ j\neq i}}^{n_1+n_2+n_3} \left(\frac{1}{(x_1^{(i)}-x_1^{(j)})^2+(x_2^{(i)}-x_2^{(j)})^2+(x_3^{(i)}-x_3^{(j)})^2} \right)^{50} \right)^{\frac{1}{50}} \\ &+ \left(\sum_{i=n_1+1}^{n_1+n_2+n_3} \left(\frac{1}{(x_2^{(i)}-0.4)^2+(x_3^{(i)}-0.5)^2} \right)^{50} \right)^{\frac{1}{50}}. \end{aligned}$$

The first part is the generalized maximin applied to $\widetilde{\mathbb{X}}_2$, the second part ensures that $\widetilde{\mathbb{X}}_2$ is far from the subspace where Z_2 is null.

– The last column $\mathbb{X}_{3,I_3}$ ($I_3 = \{4\}$ is the index set of the variables released at step 3) is finally built to ensure the properties of $\widetilde{\mathbb{X}}_3 = \mathbb{X}_3$. The goal is to create the design $\mathbb{X}_3$ such that it is nested in $\mathbb{X}_{3,I_1\cup I_2}$. $\mathbb{X}_{3,I_3}$ must minimize:

$$\begin{aligned} \Phi_{3,3}(\mathbb{X}_{3,I_3}) &= \left(\sum_{\substack{i,j=n_1+n_2+1 \\ j\neq i}}^{n_1+n_2+n_3} \left(\frac{1}{(x_1^{(i)}-x_1^{(j)})^2+(x_2^{(i)}-x_2^{(j)})^2+(x_3^{(i)}-x_3^{(j)})^2+(x_4^{(i)}-x_4^{(j)})} \right)^{50} \right)^{\frac{1}{50}} \\ &+ \left(\sum_{i=n_1+n_2+1}^{n_1+n_2+n_3} \left(\frac{1}{(x_4^{(i)}-\frac{x_1^{(i)}+x_2^{(i)}+x_3^{(i)}}{4})^2} \right)^{50} \right)^{\frac{1}{50}}. \end{aligned}$$

The first part is the generalized maximin applied to $\mathbb{X}_3$, the second part ensures that $\mathbb{X}_3$ is far from the subspace where Z_3 is null.

To illustrate the building algorithm, it is assumed that 3 non-nested designs $\mathbb{X}_1$, $\mathbb{X}_2$, $\mathbb{X}_3$ respectively of size 2, 3, and 4, must be created.

- At step 1, $\mathbb{X}_1$ is taken equal to $\mathbb{X}_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

- At step 2, the goal is to create $\mathbb{X}_2$ of size 3.

  – The first column $\mathbb{X}_{2,I_1}$ is adjusted by hand to have a good value of $\Phi_{2,1}$. $\mathbb{X}_{2,I_1}$ is

taken equal to $\mathbb{X}_{2,I_1} = \begin{pmatrix} \frac{2}{8} \\ \\ \frac{4}{8} \\ \\ \frac{6}{8} \end{pmatrix}$. The resulting $\overset{\sim}{\mathbb{X}}{}_1^2 = \begin{pmatrix} \frac{0}{8} \\ \\ \frac{8}{8} \\ \\ \frac{2}{8} \\ \\ \frac{4}{8} \\ \\ \frac{6}{8} \end{pmatrix}$ (the part of $\overset{\sim}{\mathbb{X}}_1$ concerned at step 2) is formed of equispaced points in $[0, 1]$.

– $\mathbb{X}_2$ is created as a design nested in $\mathbb{X}_{2,I_1}$. It is initialized to

$$\mathbb{X}_2 = \begin{pmatrix} \frac{2}{8} & \frac{1}{6} & \frac{1}{6} \\ \\ \frac{4}{8} & \frac{3}{6} & \frac{3}{6} \\ \\ \frac{6}{8} & \frac{5}{6} & \frac{5}{6} \end{pmatrix}, \ \Phi_{2,2}(\mathbb{X}_2) = 11.90.$$

The optimal design for the criterion $\Phi_{2,2}$ is obtained for the permutation $\begin{pmatrix} 3 & 1 & 2 \end{pmatrix}$ in the second column and the permutation $\begin{pmatrix} 2 & 3 & 1 \end{pmatrix}$ in the third column

$$\mathbb{X}_2 = \begin{pmatrix} \frac{2}{8} & \frac{5}{6} & \frac{3}{6} \\ \\ \frac{4}{8} & \frac{1}{6} & \frac{5}{6} \\ \\ \frac{6}{8} & \frac{3}{6} & \frac{1}{6} \end{pmatrix}, \ \Phi_{2,2}(\mathbb{X}_2) = 4.33.$$

It is found using algorithm 1 in appendix 9.4.2.

- At step 3, the goal is to create $\mathbb{X}_3$ of size 4.

  – The first column $\mathbb{X}_{3,I_1}$ is adjusted by hand to have a good value of $\Phi_{3,1}$. $\mathbb{X}_{3,I_1}$ is taken equal to $\mathbb{X}_{3,I_1} = \begin{pmatrix} \frac{1}{8} \\ \\ \frac{3}{8} \\ \\ \frac{5}{8} \\ \\ \frac{7}{8} \end{pmatrix}$. The resulting $\overset{\sim}{\mathbb{X}}_1 = \begin{pmatrix} \frac{0}{8} \\ \\ \frac{8}{8} \\ \\ \frac{2}{8} \\ \\ \frac{4}{8} \\ \\ \frac{6}{8} \\ \\ \frac{1}{8} \\ \\ \frac{3}{8} \\ \\ \frac{5}{8} \\ \\ \frac{7}{8} \end{pmatrix}$ is formed of equispaced points in $[0, 1]$.

– The submatrix $\mathbb{X}_{3,I_1\cup I_2}$ composed of the first three columns of $\mathbb{X}_3$ is created as a design nested in $\mathbb{X}_{3,I_1}$. It is initialized to:

$$\mathbb{X}_{3,I_1\cup I_2} = \begin{pmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \\ \frac{3}{8} & \frac{3}{8} & \frac{3}{8} \\ \frac{5}{8} & \frac{5}{8} & \frac{5}{8} \\ \frac{7}{8} & \frac{7}{8} & \frac{7}{8} \end{pmatrix}, \ \Phi_{3,2}(\mathbb{X}_{3,2}) = 10.21.$$

The optimal design for the criterion $\Phi_{3,2}$ is obtained for the permutation $\begin{pmatrix} 2 & 3 & 1 & 4 \end{pmatrix}$ in the second column and the permutation $\begin{pmatrix} 1 & 4 & 2 & 3 \end{pmatrix}$ in the third column

$$\mathbb{X}_{3,I_1\cup I_2} = \begin{pmatrix} \frac{1}{8} & \frac{3}{8} & \frac{1}{8} \\ \frac{3}{8} & \frac{5}{8} & \frac{7}{8} \\ \frac{5}{8} & \frac{1}{8} & \frac{3}{8} \\ \frac{7}{8} & \frac{7}{8} & \frac{5}{8} \end{pmatrix}, \ \Phi_{3,2}(\mathbb{X}_{3,2}) = 5.58.$$

It is found using algorithm 1 in appendix 9.4.2. The resulting $\widetilde{\mathbb{X}}_2 = \begin{pmatrix} \frac{2}{8} & \frac{5}{6} & \frac{3}{6} \\ \frac{4}{8} & \frac{1}{6} & \frac{5}{6} \\ \frac{6}{8} & \frac{3}{6} & \frac{1}{6} \\ \frac{1}{8} & \frac{3}{8} & \frac{1}{8} \\ \frac{3}{8} & \frac{5}{8} & \frac{7}{8} \\ \frac{5}{8} & \frac{1}{8} & \frac{3}{8} \\ \frac{7}{8} & \frac{7}{8} & \frac{5}{8} \end{pmatrix}$ has the good properties: space-filling with separate points and far from the nullity zone $(x_2, x_3) = (0.4, 0.5)$.

– $\mathbb{X}_3$ is finally created as a design nested in $\mathbb{X}_{3,I_1\cup I_2}$. It is initialized to

$$\mathbb{X}_3 = \begin{pmatrix} \frac{1}{8} & \frac{3}{8} & \frac{1}{8} & \frac{1}{8} \\ \frac{3}{8} & \frac{5}{8} & \frac{7}{8} & \frac{3}{8} \\ \frac{5}{8} & \frac{1}{8} & \frac{3}{8} & \frac{5}{8} \\ \frac{7}{8} & \frac{7}{8} & \frac{5}{8} & \frac{7}{8} \end{pmatrix}, \ \Phi_{3,3}(\mathbb{X}_2) = 33.29.$$

The optimal design for the criterion $\Phi_{3,3}$ is obtained for the permutation $\begin{pmatrix} 2 & 1 & 3 & 4 \end{pmatrix}$

in the last column:

$$\mathbb{X}_3 = \begin{pmatrix} \frac{1}{8} & \frac{3}{8} & \frac{1}{8} & \frac{3}{8} \\ \frac{3}{8} & \frac{5}{8} & \frac{7}{8} & \frac{1}{8} \\ \frac{5}{8} & \frac{1}{8} & \frac{3}{8} & \frac{5}{8} \\ \frac{7}{8} & \frac{7}{8} & \frac{5}{8} & \frac{7}{8} \end{pmatrix}, \ \Phi_{3,3}(\mathbb{X}_2) = 6.08.$$

It is found using algorithm 1 in appendix 9.4.2.

9.1.4 Illustration of chapter 6

Multi-conditioning The example in 4D is modified to take into account multiple subspaces. The output considered is still a function of 4 inputs $f(x_1, x_2, x_3, x_4)$. The study is composed of 3 steps with $I_1 = \{1\}$, $I_2 = \{2, 3\}$ and $I_3 = \{4\}$.

- At step 1, x_1 is released and (x_2, x_3, x_4) are fixed. Three different restrictions are considered: $f(x_1, 0.4, 0.5, \frac{x_1+0.4+0.5}{4})$, $f(x_1, 0.1, 0.7, \frac{x_1+0.1+0.7}{4})$, and $f(x_1, 0.8, 0.3, \frac{x_1+0.8+0.3}{4})$. On the three subspaces considered, the values of (x_2, x_3) are modified by an additive constant:

$$\begin{cases} \grave{x}_{I_2} &= (0.4, 0.5), \\ \mathbb{B}_2 &= \begin{pmatrix} 0 & 0 \\ -0.3 & 0.2 \\ 0.4 & -0.2 \end{pmatrix}. \end{cases}$$

  and x_4 is equal to the same function of (x_1, x_2, x_3): $\grave{x}_4 = \frac{x_1+x_2+x_3}{4}$. The restrictions can be summarized in the notation $f(x_{I_1}, \grave{x}_{I_2} + \mathbb{B}_2, \grave{x}_{I_3})$. The three restrictions are modeled by $Y_1(x_1, x_2, x_3) = m + Z_1(x_1, x_2, x_3)$, such that $f(x_1, 0.4, 0.5, \frac{x_1+0.4+0.5}{4})$ is the realization of $Y_1(x_1, 0.4, 0.5)$, $f(x_1, 0.1, 0.7, \frac{x_1+0.1+0.7}{4})$ is the realization of $Y_2(x_1, 0.1, 0.7)$, and $f(x_1, 0.8, 0.3, \frac{x_1+0.8+0.3}{4})$ is the realization of $Y_3(x_1, 0.8, 0.3)$. This can be summarized by saying that $f(x_{I_1}, \grave{x}_{I_2} + \mathbb{B}_2, \grave{x}_{I_3})$ is the realization of $Y_1(x_{I_1}, \grave{x}_{I_2} + \mathbb{B}_2)$. Z_1 is a centered Gaussian process of covariance kernel $\sigma_1^2 \rho_1((x_1, x_2, x_3), (t_1, t_2, t_3))$. Three DoE's are generated $\mathbb{X}_1^1, \mathbb{X}_1^2, \mathbb{X}_1^3 \subset [0, 1]^3$, of the form:

$$\mathbb{X}_1^1 = \begin{pmatrix} x_1^{(1,1)} & 0.4 & 0.5 \\ \vdots & \vdots & \vdots \\ x_1^{(1,n_1^1)} & 0.4 & 0.5 \end{pmatrix}, \quad \mathbb{X}_1^2 = \begin{pmatrix} x_1^{(1,n_1^1+1)} & 0.1 & 0.7 \\ \vdots & \vdots & \vdots \\ x_1^{(1,n_1^1+n_1^2)} & 0.1 & 0.7 \end{pmatrix}, \quad \mathbb{X}_1^3 = \begin{pmatrix} x_1^{(1,n_1^1+n_1^2+1)} & 0.8 & 0.3 \\ \vdots & \vdots & \vdots \\ x_1^{(1,n_1^1+n_1^2+n_1^3)} & 0.8 & 0.3 \end{pmatrix},$$

  with $\boldsymbol{y_1^1}$, $\boldsymbol{y_1^2}$, $\boldsymbol{y_1^3}$ the corresponding output values.

- At step 2, (x_2, x_3) are released and x_4 is still fixed. Two different restrictions are considered: $f(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4})$ and $f(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4} + \frac{1}{4})$. The first subspace has the same x_4 value than at step 1, the value of x_4 on the second subspace is modified by an additive constant:

$$\begin{cases} \grave{x}_{I_3} &= \frac{x_1+x_2+x_3}{4}, \\ \mathbb{B}_3 &= \begin{pmatrix} 0 \\ \frac{1}{4} \end{pmatrix}. \end{cases}$$

  The restrictions can be summarized in the notation $f(x_{I_1}, x_{I_2}, \grave{x}_{I_3} + \mathbb{B}_3)$. The two restrictions are modeled by $Y_2(x_1, x_2, x_3, x_4) = Y_1(x_1, x_2, x_3) + Z_2(x_1, x_2, x_3, x_4)$, such that

$f(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4})$ is the realization of $Y_2(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4})$, and $f(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4} + \frac{1}{4})$ is the realization of $Y_2(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4} + \frac{1}{4})$. This can be summarized by saying that $f(x_{I_1}, x_{I_2}, \grave{x}_{I_3} + \mathbb{B}_3)$ is the realization of $Y_2(x_{I_1}, x_{I_2}, \grave{x}_{I_3} + \mathbb{B}_3)$. Z_2 is a centered Gaussian process of covariance kernel $\sigma_2^2 \rho_2((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4))$ such that:

$$\left\{ \begin{array}{rcll} Z_2(x_1, 0.4, 0.5, \frac{x_1+0.4+0.5}{4}) & = & 0, & \forall x_1 \in [0,1], \\ Z_2(x_1, 0.1, 0.7, \frac{x_1+0.1+0.7}{4}) & = & 0, & \forall x_1 \in [0,1], \\ Z_2(x_1, , 0.8, 0.3, \frac{x_1+0.8+0.3}{4}) & = & 0, & \forall x_1 \in [0,1]. \end{array} \right.$$

This can be summarized in: $Z_2(x_{I_1}, \grave{x}_{I_2} + \mathbb{B}_2, \grave{x}_{I_3}) = 0$, for all x_1 in $[0,1]$. Two DoE's are generated $\mathbb{X}_2^1, \mathbb{X}_2^2 \subset [0,1]^4$, of the form:

$$\left\{ \begin{array}{rcl} \mathbb{X}_2^1 & = & \begin{pmatrix} x_1^{(2,1)} & x_2^{(2,1)} & x_3^{(2,1)} & \frac{x_1^{(2,1)}+x_2^{(2,1)}+x_3^{(2,1)}}{4} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(2,n_2^1)} & x_2^{(2,n_2^1)} & x_3^{(2,n_2^1)} & \frac{x_1^{(2,n_2^1)}+x_2^{(2,n_2^1)}+x_3^{(2,n_2^1)}}{4} \end{pmatrix}, \\ \mathbb{X}_2^2 & = & \begin{pmatrix} x_1^{(2,n_2^1+1)} & x_2^{(2,n_2^1+1)} & x_3^{(2,n_2^1+1)} & \frac{x_1^{(2,n_2^1+1)}+x_2^{(2,n_2^1+1)}+x_3^{(2,n_2^1+1)}}{4} + \frac{1}{4} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(2,n_2^1+n_2^2)} & x_2^{(2,n_2^1+n_2^2)} & x_3^{(2,n_2^1+n_2^2)} & \frac{x_1^{(2,n_2^1+n_2^2)}+x_2^{(2,n_2^1+n_2^2)}+x_3^{(2,n_2^1+n_2^2)}}{4} + \frac{1}{4} \end{pmatrix}, \end{array} \right.$$

with $\boldsymbol{y_2^1}$, $\boldsymbol{y_2^2}$ the corresponding output values.

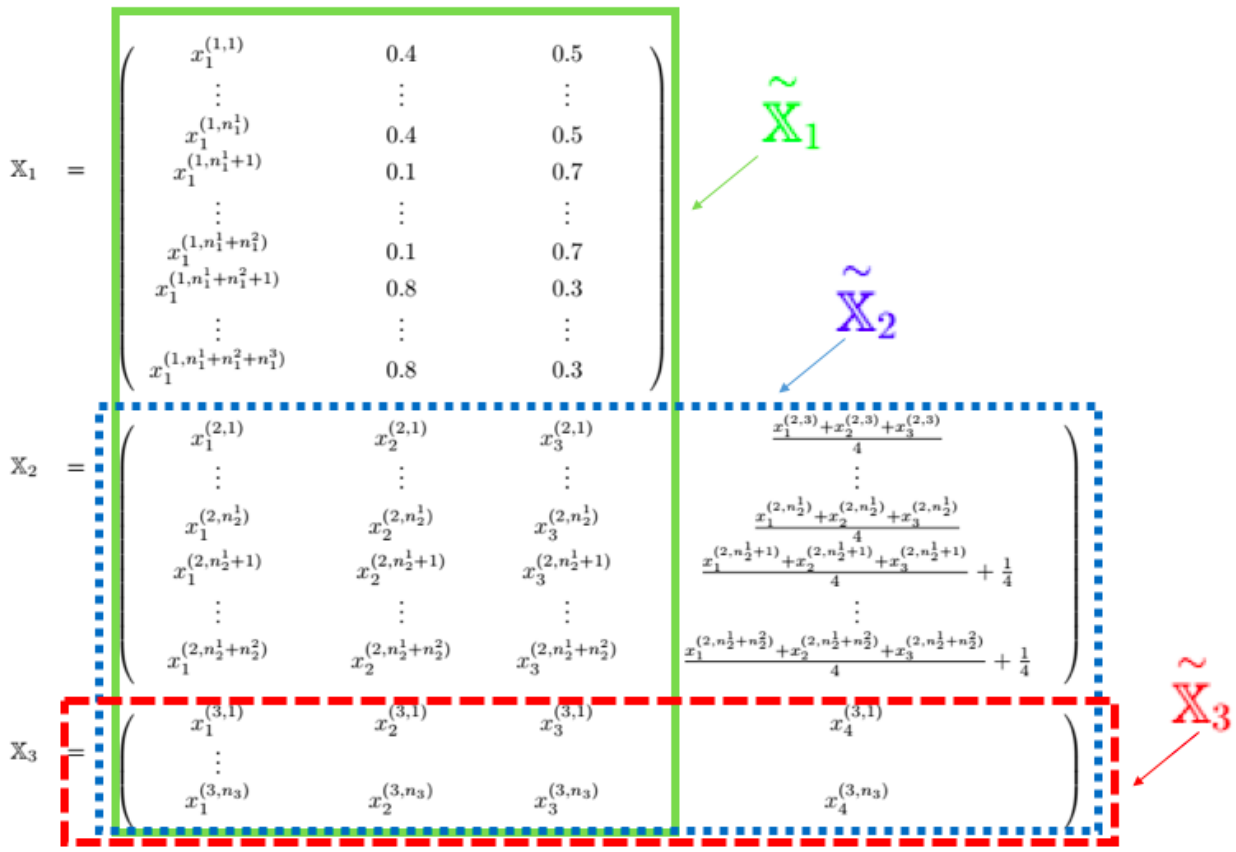
- At step 3, x_4 is released. f is modeled as the realization of Y_3 which is defined by $Y_3(x_1, x_2, x_3, x_4) = Y_2(x_1, x_2, x_3, x_4) + Z_3(x_1, x_2, x_3, x_4)$. Z_3 is a centered Gaussian process of covariance kernel $\sigma_3^2 \rho_3((x_1, x_2, x_3, x_4), (t_1, t_2, t_3, t_4))$ such that:

$$\left\{ \begin{array}{rcll} Z_3(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}) & = & 0, & \forall (x_1, x_2, x_3) \in [0,1]^3, \\ Z_3(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4} + \frac{1}{4}) & = & 0, & \forall (x_1, x_2, x_3) \in [0,1]^3. \end{array} \right.$$

This can be summarized in: $Z_3(x_{I_1}, x_{I_2}, \grave{x}_{I_3} + \mathbb{B}_3) = 0$, for all (x_{I_1}, x_{I_2}) in $[0,1]^3$. One DoE $\mathbb{X}_3 \subset [0,1]^4$ is generated with the corresponding output value $\boldsymbol{y_3}$:

$$\mathbb{X}_3 = \begin{pmatrix} x_1^{(3,1)} & x_2^{(3,1)} & x_3^{(3,1)} & x_4^{(3,1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(3,n_3)} & x_2^{(3,n_3)} & x_3^{(3,n_3)} & x_4^{(3,n_3)} \end{pmatrix}.$$

The new definition of $\widetilde{\mathbb{X}}_1$, $\widetilde{\mathbb{X}}_2$, and $\widetilde{\mathbb{X}}_3$ is shown on figure below.



The goal is now to find candidates for the correction processes.

- At step 2, $J = \{1\}$, $I = \{2, 3, 4\}$. $Z_2(x_1, x_2, x_3, x_4)$ must be null on:

$$\mathcal{D}_2 = \{(s_1, g_2(s_1) + (s_2, s_3, s_4)), s = (s_1, s_2, s_3, s_4) \in \underbrace{[0,1] \times \mathcal{B}_2}_{=\mathcal{S}_2}\},$$

with

$$\begin{cases} g_2(s_1) &= (0.4, 0.5, \frac{s_1+0.4+0.5}{4}), & \forall s_1 \in [0,1], \\ \mathcal{B}_2 &= \begin{pmatrix} 0 & 0 & 0 \\ -0.3 & 0.2 & \frac{-0.3+0.2}{4} \\ 0.4 & -0.2 & \frac{0.4-0.2}{4} \end{pmatrix}. \end{cases}$$

Z_2 is defined as:

$$Z_2(x_1, x_2, x_3, x_4) = \widetilde{Z}_2(x_1, x_2, x_3, x_4) - \mathbb{E}\left[\widetilde{Z}_2(x_1, x_2, x_3, x_4) \mid \widetilde{Z}_2(\mathcal{D}_2)\right],$$

with $\widetilde{Z}_2$ a centered Gaussian process of covariance kernel equal to $\sigma_2^2 r_{2,1}(x_1, t_1) r_{2,2}((x_2, x_3, x_4) - g_2(x_1), (t_2, t_3, t_4) - g_2(t_1))$ and $r_{2,1}$, $r_{2,2}$ stationary correlation kernels. It is equal to:

$$\begin{aligned} Z_2(x_1, x_2, x_3, x_4) &= \widetilde{Z}_2(x_1, x_2, x_3, x_4) \\ &- r_{2,2}\left((x_2, x_3, x_4) - g(x_1), \mathcal{B}_2\right) r_{2,2}\left(\mathcal{B}_2, \mathcal{B}_2\right)^{-1} \widetilde{Z}_2(x_1, g(x_1) + \mathcal{B}_2). \end{aligned}$$

- At step 3, $J = \{1, 2, 3\}$, $I = \{4\}$. Z_3 must be null on:

$$\mathcal{D}_3 = \{(s_1, s_2, s_3, g_3(s_1, s_2, s_3) + s_4), (s_1, s_2, s_3, s_4) \in \underbrace{[0, 1]^3 \times \mathcal{B}_3}_{=\mathcal{S}_3}\},$$

with

$$\begin{cases} g_3(x_1, x_2, x_3) & = & \frac{x_1+x_2+x_3}{4}, \quad \forall (x_1, x_2, x_3) \in [0, 1]^3 \\ \mathcal{B}_3 & = & \begin{pmatrix} 0 \\ \frac{1}{4} \end{pmatrix}. \end{cases}$$

Z_3 is defined as:

$$Z_3(x_1, x_2, x_3, x_4) = \widetilde{Z}_3(x_1, x_2, x_3, x_4) - \mathbb{E}\left[\widetilde{Z}_3(x_1, x_2, x_3, x_4) \mid \widetilde{Z}_3(\mathcal{D}_3)\right],$$

with $\widetilde{Z}_3$ a centered Gaussian process of covariance kernel equal to $\sigma_3^2 r_{3,1}((x_1, x_2, x_3), (t_1, t_2, t_3)) r_{3,2}(x_4 - g_3(x_1, x_2, x_3), t_4 - g_3(t_1, t_2, t_3))$ and $r_{3,1}$, $r_{3,2}$ stationary correlation kernels.It is equal to:

$$\begin{aligned} &Z_3(x_1, x_2, x_3, x_4) = \widetilde{Z}_3(x_1, x_2, x_3, x_4) \\ &- r_{3,2}\left(x_4 - g_3(x_1, x_2, x_3), \mathcal{B}_3\right) r_{3,2}\left(\mathcal{B}_3, \mathcal{B}_3\right)^{-1} \widetilde{Z}_3(x_1, x_2, x_3, g_3(x_1, x_2, x_3) + \mathcal{B}_3)). \end{aligned}$$

Enrichment In the example in 4D from subsection 3.1.1 in chapter 3, the metamodel seqGPR has been built on the non-nested samples:

$$\begin{cases} \mathbb{X}_1 = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1)} \end{pmatrix}, \\ \mathbb{X}_2 = \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2)} & x_2^{(n_1+n_2)} & x_3^{(n_1+n_2)} \end{pmatrix}, \\ \mathbb{X}_3 = \begin{pmatrix} x_1^{(n_1+n_2+1)} & x_2^{(n_1+n_2+1)} & x_3^{(n_1+n_2+1)} & x_4^{(n_1+n_2+1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2+n_3)} & x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} & x_4^{(n_1+n_2+n_3)} \end{pmatrix}. \end{cases}$$

with the associated output values $\boldsymbol{y_1}$, $\boldsymbol{y_2}$, and $\boldsymbol{y_3}$. The designs $\widetilde{\mathbb{X}}_1$ and $\widetilde{\mathbb{X}}_2$ are defined as:

$$\begin{cases} \widetilde{\mathbb{X}}_1 & = & \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_1^{(n_1+n_2+n_3)} \end{pmatrix}, \\ \widetilde{\mathbb{X}}_2 & = & \begin{pmatrix} x_1^{(n_1+1)} & x_2^{(n_1+1)} & x_3^{(n_1+1)} \\ \vdots & \vdots & \vdots \\ x_1^{(n_1+n_2+n_3)} & x_2^{(n_1+n_2+n_3)} & x_3^{(n_1+n_2+n_3)} \end{pmatrix}. \end{cases}$$

The enrichment point $x = (x_1, x_2, x_3, x_4)$ must maximize the following objective function which is an adjusted prediction variance taking into account the LOO-CV prediction error:

$$\tau(x) = \hat{v}(x)\left[1 + \tau_1(x) + \tau_2(x) + \tau_3(x)\right]$$

with:

$$\begin{cases} \tau_1(x) &= \displaystyle\sum_{i=1}^{n_1} \frac{\left(y_i - \hat{y}_{-i}\left(x_1^{(i)}, 0.4, 0.5, \frac{x_1^{(i)}+0.9}{4}\right)\right)^2}{\hat{v}_{-i}\left(x_1^{(i)}, 0.4, 0.5, \frac{x_1^{(i)}+0.9}{4}\right)} \mathbf{1}_{V_i}(x), \\ \tau_2(x) &= \displaystyle\sum_{i=n_1+1}^{n_1+n_2} \frac{\left(y_i - \hat{y}_{-i}\left(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}, \frac{x_1^{(i)}+x_2^{(i)}+x_3^{(i)}}{4}\right)\right)^2}{\hat{v}_{-i}\left(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}, \frac{x_1^{(i)}+x_2^{(i)}+x_3^{(i)}}{4}\right)} \mathbf{1}_{V_i}(x), \\ \tau_3(x) &= \displaystyle\sum_{i=n_1+n_2+1}^{n_1+n_2+n_3} \frac{\left(y_i - \hat{y}_{-i}\left(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}, x_4^{(i)}\right)\right)^2}{\hat{v}_{-i}\left(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}, x_4^{(i)}\right)} \mathbf{1}_{V_i}(x). \end{cases}$$

τ_1, τ_2, τ_3 are LOO-CV criteria respectively applied to $\mathbb{X}_1$, $\mathbb{X}_2$, and $\mathbb{X}_3$.

The enrichment point must verify some constraints:

$$\begin{cases} dist(x_{I_1}, \widetilde{\mathbb{X}}_1) > u, \\ dist(x_{I_2}, \grave{x}_{I_2}) > v_2, \\ dist(x_{I_3}, \grave{x}_{I_3}) > v_3. \end{cases}$$

- The constraint $dist(x_{I_1}, \widetilde{\mathbb{X}}_1) > u$ ensures that $\widetilde{\mathbb{X}}_1$, $\widetilde{\mathbb{X}}_2$ and $\mathbb{X}_3$ enriched with x are still composed of points sufficiently distant from each other. This constraint can be rewritten as:
$$\min_{1 \leqslant i \leqslant n_1+n_2+n_3} \left| x_1 - x_1^{(i)} \right| > u.$$

- The constraint $dist(x_{I_2}, \grave{x}_{I_2}) > v_2$ ensures that $\widetilde{\mathbb{X}}_2$ enriched with x is still far from the nullity subspace $\mathcal{D}_2 = \{(x_1, 0.4, 0.5), x_1 \in [0,1]\}$. This constraint can be rewritten as:
$$\sqrt{(x_2 - 0.4)^2 + (x_3 - 0.5)^2} > v_2.$$

- The constraint $dist(x_{I_3}, \grave{x}_{I_3}) > v_3$ ensures that $\mathbb{X}_3$ enriched with x is still far from the nullity subspace $\mathcal{D}_3 = \left\{(x_1, x_2, x_3, \frac{x_1+x_2+x_3}{4}), (x_1, x_2, x_3) \in [0,1]^3\right\}$. This constraint can be rewritten as:
$$\left| x_4 - \frac{x_1 + x_2 + x_3}{4} \right| > v_3.$$

The optimization problem is then equal to:

$$\min_{x \in [0,1]^4} \tau(x),$$

$$\left| \begin{array}{l} \displaystyle\min_{1 \leqslant i \leqslant n_1+n_2+n_3} \left| x_1 - x_1^{(i)} \right| > u, \\ \\ \sqrt{(x_2 - 0.4)^2 + (x_3 - 0.5)^2} > v_2, \\ \\ \left| x_4 - \frac{x_1+x_2+x_3}{4} \right| > v_3. \end{array} \right.$$

9.2 Proofs state-of-the-art

This section gives the proofs of the propositions of chapter 2 which draws up a state of the art of the tools linked to the thesis. As a recall, all equalities between processes are in $L^2(\Omega)$ (see definition 1):

$$X = Y \;\Leftrightarrow\; \mathbb{E}\left[(X-Y)^2\right] = 0.$$

9.2.1 Proof of proposition 1

This subsection deals with the proof of proposition 1, which gives an explicit formula of the conditional expectation in the Gaussian case for any covariance matrix Σ_{22}. As a reminder, $\boldsymbol{V} = (\boldsymbol{V_1}, \boldsymbol{V_2})$ is a Gaussian vector of mean vector and covariance matrix given by:

$$\begin{cases} \mathbb{E}[\boldsymbol{V}] &= \begin{pmatrix} \boldsymbol{m_1} \\ \boldsymbol{m_2} \end{pmatrix}, \\ Cov(\boldsymbol{V}, \boldsymbol{V}) &= \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}, \end{cases}$$

where $\boldsymbol{m_1} = \mathbb{E}[\boldsymbol{V_1}]$, $\boldsymbol{m_2} = \mathbb{E}[\boldsymbol{V_2}]$, $\Sigma_{11} = Cov(\boldsymbol{V_1}, \boldsymbol{V_1})$, $\Sigma_{12} = Cov(\boldsymbol{V_1}, \boldsymbol{V_2})$, $\Sigma_{21} = Cov(\boldsymbol{V_2}, \boldsymbol{V_1})$, and $\Sigma_{22} = Cov(\boldsymbol{V_2}, \boldsymbol{V_2})$.

Proof *Let* $\boldsymbol{V_1} = \left(V_1^{(1)}, \cdots, V_1^{(m)}\right)'$ *and* $\boldsymbol{V}_2 = \left(V_2^{(1)}, \cdots, V_2^{(n)}\right)'$ *denote the components of* $\boldsymbol{V_1}$ *and* $\boldsymbol{V_2}$. $\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]$ *is easier to compute component by component, i.e.* $\mathbb{E}\left[V_1^{(i)} \mid \boldsymbol{V_2}\right]$.

- *First case:* $\boldsymbol{m_1} = \boldsymbol{m_2} = 0$. *The goal is to find* $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_n)$ *solution of*

$$\min_{\alpha} \mathbb{E}\left[\left(V_1^{(i)} - \boldsymbol{\alpha}'\boldsymbol{V_2}\right)^2\right]$$

Then the conditional expectation is equal to:

$$\mathbb{E}\left[V_1^{(i)} \mid \boldsymbol{V_2}\right] = \boldsymbol{\alpha}'\boldsymbol{V_2}.$$

The expectation to minimize can be rewritten as

$$\mathbb{E}\left[\left(V_1^{(i)} - \boldsymbol{\alpha}'\boldsymbol{V_2}\right)^2\right] \;=\; Var\left(V_1^{(i)}\right) + \boldsymbol{\alpha}'\Sigma_{22}\boldsymbol{\alpha} - 2\boldsymbol{\alpha}'Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right). \tag{9.1}$$

As Σ_{22} *is a covariance matrix, it is positive semidefinite and can be decomposed as:*

$$\Sigma_{22} = PSP' \tag{9.2}$$

with $P \times P' = I$ *and* $S = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & & \vdots \\ 0 & \dots & 0 & \lambda_q & 0 & \vdots & 0 \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 \\ \vdots & & & \vdots & \vdots & & \vdots \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 \end{pmatrix}$ *where* $(\lambda_1, \cdots, \lambda_q)$ *are the positive eigen values of* Σ_{22}. *Then, injecting the spectral decomposition of* Σ_{22} *(equation*

(9.2)) *in the expectation (equation* (9.1)) *and using* $PP' = I$*, it becomes:*

$$\begin{aligned}\mathbb{E}\left[\left(\boldsymbol{\alpha}'\boldsymbol{V_2} - V_1^{(i)}\right)^2\right] &= Var\left(V_1^{(i)}\right) + \boldsymbol{\alpha}'PSP'\boldsymbol{\alpha} - 2\boldsymbol{\alpha}'P \times P'Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right) \\ &= Var\left(V_1^{(i)}\right) + \boldsymbol{x}'S\boldsymbol{x} - \boldsymbol{x}'\boldsymbol{c},\end{aligned}$$

with $\boldsymbol{x} = P'\boldsymbol{\alpha}$ *and* $\boldsymbol{c} = 2P'Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right)$*. Then, denoting* $\boldsymbol{x} = (x_1, \cdots, x_n)'$ *and* $\boldsymbol{c} = (c_1, \cdots, c_n)'$*, the goal is to find* $\boldsymbol{x}$ *minimizing*

$$\sum_{j=1}^{q} \lambda_j x_j^2 - \sum_{j=1}^{n} c_j x_j$$

- *If* $q = n$ *(i.e.* Σ_{22} *is positive definite), then* $\boldsymbol{x}$ *must minimize* $\sum_{j=1}^{n} \lambda_j x_j^2 - c_j x_j$*. So each* x_j *must minimize a quadratic polynomial* $\lambda_j x_j^2 - c_j x_j$*. The solution is* $x_j = \frac{c_j}{2\lambda_j}$*, or*

$$\begin{aligned}\boldsymbol{x} &= \begin{pmatrix} \frac{c_1}{2\lambda_1} \\ \vdots \\ \frac{c_n}{2\lambda_n} \end{pmatrix} \\ &= \tfrac{1}{2}S^{-1}\boldsymbol{c} \\ &= S^{-1}P'Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right).\end{aligned}$$

In this case

$$\begin{aligned}\alpha &= P\boldsymbol{x} \\ &= PS^{-1}P'Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right) \\ &= \Sigma_{22}^{-1}Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right).\end{aligned}$$

- *If* $q < n$*, then* $\boldsymbol{x}$ *must minimize* $\sum_{j=1}^{q}(\lambda_j x_j^2 - c_j x_j) - \sum_{j=q+1}^{n} c_j x_j$*. In fact, for all* $j \geqslant q+1$*,* $c_j = 0$*. Indeed, if there exists* $j \geqslant q+1$ *such that* $c_j \neq 0$*, then by*

taking $\boldsymbol{x} = \begin{matrix} 1 \\ \vdots \\ j-1 \\ j \\ j+1 \\ \vdots \\ n \end{matrix} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ c_j + \frac{Var(V_1^{(i)})}{c_j} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ *(or the corresponding* $\boldsymbol{\alpha} = P'\boldsymbol{x}$*), the initial expectation to minimize becomes*

$$\begin{aligned}\mathbb{E}\left[\left(\boldsymbol{\alpha}'\boldsymbol{V_2} - V_1^{(i)}\right)^2\right] &= Var\left(V_1^{(i)}\right) + \boldsymbol{x}'S\boldsymbol{x} - \boldsymbol{x}'\boldsymbol{c} \\ &= Var\left(V_1^{(i)}\right) + 0 - c_j^2 - Var\left(V_1^{(i)}\right) \\ &= -c_j^2 < 0\end{aligned}$$

which is impossible (the expectation of a square in non-negative). This time, $\boldsymbol{x}$

must minimize $\sum_{j=1}^{q} \lambda_j x_j^2 - c_j x_j$. *There is apparently an infinite number of solutions* $\boldsymbol{x} \in \left\{ \begin{pmatrix} \frac{c_1}{2\lambda_1} \\ \vdots \\ \frac{c_q}{2\lambda_q} \\ x_{q+1} \\ \vdots \\ x_n \end{pmatrix}, (x_{q+1}, \cdots, x_n) \in \mathbb{R}^{n-q} \right\}$,

or $\alpha \in \left\{ P \begin{pmatrix} \frac{c_1}{2\lambda_1} \\ \vdots \\ \frac{c_q}{2\lambda_q} \\ x_{q+1} \\ \vdots \\ x_n \end{pmatrix}, (x_{q+1}, \cdots, x_n) \in \mathbb{R}^{n-q} \right\}$, *but the corresponding random variables are all equal in* $L^2(\Omega)$. *Indeed:*

$$\begin{aligned}
&\mathbb{E}\left[\left(\begin{pmatrix} \frac{c_1}{2\lambda_1} & \cdots & \frac{c_q}{2\lambda_q} & x_{q+1} & \cdots & x_n \end{pmatrix} P' \boldsymbol{V_2} - \begin{pmatrix} \frac{c_1}{2\lambda_1} & \cdots & \frac{c_p}{2\lambda_q} & t_{q+1} & \cdots & t_n \end{pmatrix} P' \boldsymbol{V_2}\right)^2\right] \\
&= \mathbb{E}\left[\left(\begin{pmatrix} 0 & \cdots & 0 & x_{q+1} & \cdots & x_n \end{pmatrix} P' \boldsymbol{V_2} - \begin{pmatrix} 0 & \cdots & 0 & t_{q+1} & \cdots & t_n \end{pmatrix} P' \boldsymbol{V_2}\right)^2\right] \\
&= \mathbb{E}\left[\left(\begin{pmatrix} 0 & \cdots & 0 & x_{q+1} - t_{q+1} & \cdots & x_n - t_n \end{pmatrix} P' \boldsymbol{V_2}\right)^2\right] \\
&= \begin{pmatrix} 0 & \cdots & 0 & x_{q+1} - t_{q+1} & \cdots & x_n - t_n \end{pmatrix} \underbrace{P'\Sigma_{22}P}_{=S} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ x_{q+1} - t_{q+1} \\ \vdots \\ x_n - t_n \end{pmatrix} \\
&= 0
\end{aligned}$$

In particular, the simplest solution is:

$$\begin{aligned}
\boldsymbol{\alpha} &= P \begin{pmatrix} \frac{c_1}{2\lambda_1} \\ \vdots \\ \frac{c_q}{2\lambda_q} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\
&= \frac{1}{2} P \begin{pmatrix} \frac{1}{\lambda_1} & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & \vdots & 0 & \vdots & \cdots & \vdots \\ 0 & \cdots & \frac{1}{\lambda_q} & 0 & \cdots & 0 & \\ 0 & \cdots & 0 & 0 & \cdots & 0 & \\ \vdots & & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 & \end{pmatrix} \boldsymbol{c} \\
&= PS^{+}P'Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right) \\
&= \Sigma_{22}^{+} Cov\left(\boldsymbol{V_2}, V_1^{(i)}\right)
\end{aligned}$$

The conditional expectation is equal to:

$$\begin{aligned}
\mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right] &= \begin{pmatrix} \mathbb{E}\left[V_1^{(1)} \mid \boldsymbol{V_2}\right] \\ \vdots \\ \mathbb{E}\left[V_1^{(m)} \mid \boldsymbol{V_2}\right] \end{pmatrix} \\
&= \begin{pmatrix} Cov\left(V_1^{(1)}, \boldsymbol{V_2}\right)\Sigma_{22}^{+}\boldsymbol{V_2} \\ \vdots \\ Cov\left(V_1^{(m)}, \boldsymbol{V_2}\right)\Sigma_{22}^{+}\boldsymbol{V_2} \end{pmatrix} \\
&= \begin{pmatrix} Cov\left(V_1^{(1)}, \boldsymbol{V_2}\right) \\ \vdots \\ Cov\left(V_1^{(m)}, \boldsymbol{V_2}\right) \end{pmatrix}\Sigma_{22}^{+}\boldsymbol{V_2} \\
&= \Sigma_{12}\Sigma_{22}^{+}\boldsymbol{V_2}
\end{aligned}$$

- *Second case: $\boldsymbol{m_1} \neq 0$, $\boldsymbol{m_2} \neq 0$. $\boldsymbol{V_1} - \boldsymbol{m_1}$ and $\boldsymbol{V_2} - \boldsymbol{m_2}$ are centered Gaussian vectors. The previous formula can be used:*

$$\mathbb{E}\left[\boldsymbol{V_1} - \boldsymbol{m_1} \mid \boldsymbol{V_2} - \boldsymbol{m_2}\right] = \Sigma_{12}\Sigma_{22}^{+}\left(\boldsymbol{V_2} - \boldsymbol{m_2}\right).$$

As the conditional expectation is linear and the event spaces generated by $\boldsymbol{V_2}$ and $\boldsymbol{V_2} - \boldsymbol{m_2}$ are equal, it follows that:

$$\mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right] = \boldsymbol{m_1} + \Sigma_{12}\Sigma_{22}^{+}\left(\boldsymbol{V_2} - \boldsymbol{m_2}\right).$$

The mean of $\mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right]$ is $\boldsymbol{m_1}$ and its covariance matrix is:

$$\begin{aligned}
Cov\left(\mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right], \mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right]\right) &= Cov\left(\Sigma_{12}\Sigma_{22}^{+}\boldsymbol{V_2}, \Sigma_{12}\Sigma_{22}^{+}\boldsymbol{V_2}\right) \\
&= \Sigma_{12}\underbrace{\Sigma_{22}^{+}\Sigma_{22}\Sigma_{22}^{+}}_{=\Sigma_{22}^{+}}\Sigma_{21} \\
&= \Sigma_{12}\Sigma_{22}^{+}\Sigma_{21}.
\end{aligned}$$

■

9.2.2 Proof of proposition 2

This section deals with the proof of proposition 2, which gives a formula of the conditional vector $[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}]$, following the notations of subsection 9.2.1.

Proof • *First case: $\boldsymbol{m_1} = \boldsymbol{m_2} = 0$. As for all i in $[\![1, m]\!]$, $\mathbb{E}\left[V_1^{(i)} \mid \boldsymbol{V_2}\right]$ is the orthogonal projection $V_1^{(i)}$ in $span(\boldsymbol{V_2})$, $\boldsymbol{V_1}$ can be decomposed as*

$$\begin{aligned}
\boldsymbol{V_1} &= \mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right] + \left(\boldsymbol{V_1} - \mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right]\right) \\
&= \Sigma_{12}\Sigma_{22}^{+}\boldsymbol{V_2} + \left(\boldsymbol{V_1} - \mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right]\right)
\end{aligned} \tag{9.3}$$

with for all i in $[\![1, m]\!]$, $V_1^{(i)} - \mathbb{E}\left[V_1^{(i)} \mid \boldsymbol{V_2}\right] \perp span(\boldsymbol{V_2})$. Then conditioning by $\boldsymbol{V_2} = \boldsymbol{v}$ in equation (9.3):

$$\left[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}\right] = \Sigma_{12}\Sigma_{22}^{+}\boldsymbol{v} + \left(\boldsymbol{V_1} - \mathbb{E}\left[\boldsymbol{V_1} \mid \boldsymbol{V_2}\right]\right). \tag{9.4}$$

For all i in $[\![1, m]\!]$, $V_1^{(i)} - \mathbb{E}\left[V_1^{(i)} \mid \boldsymbol{V_2}\right]$ is unchanged because of its orthogonality to $span(\boldsymbol{V_2})$.

- *Second case:* $\boldsymbol{m_1} \neq 0$, $\boldsymbol{m_2} \neq 0$. *The previous formula can be applied to* $\boldsymbol{V_1} - \boldsymbol{m_1}$ *and* $\boldsymbol{V_2} - \boldsymbol{m_2}$*:*

$$\begin{array}{ll} & [\boldsymbol{V_1} - \boldsymbol{m_1} \mid \boldsymbol{V_2} - \boldsymbol{m_2} = \boldsymbol{v} - \boldsymbol{m_2}] = \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{v} - \boldsymbol{m_2}) + (\boldsymbol{V_1} - \boldsymbol{m_1} - \mathbb{E}[\boldsymbol{V_1} - \boldsymbol{m_1} \mid \boldsymbol{V_2} - \boldsymbol{m_2}]), \\ \Leftrightarrow & [\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}] - \boldsymbol{m_1} = \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{v} - \boldsymbol{m_2}) + (\boldsymbol{V_1} - \boldsymbol{m_1} - \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}] + \boldsymbol{m_1}), \\ \Leftrightarrow & [\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}] = \boldsymbol{m_1} + \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{v} - \boldsymbol{m_2}) + (\boldsymbol{V_1} - \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]), \end{array}$$

The expectation of $[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}]$ *has the same formula as* $\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]$ *but* $\boldsymbol{V_2}$ *is replaced by* $\boldsymbol{v}$*:*

$$\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}] = \boldsymbol{m_1} + \Sigma_{12}\Sigma_{22}^{+}(\boldsymbol{v} - \boldsymbol{m_2}).$$

The covariance matrix of $[\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}]$ *is equal to:*

$$\begin{array}{rcl} Cov([\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}], [\boldsymbol{V_1} \mid \boldsymbol{V_2} = \boldsymbol{v}]) & = & Cov(\boldsymbol{V_1} - \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}], \boldsymbol{V_1} - \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]) \\ & = & Cov(\boldsymbol{V_1}, \boldsymbol{V_1}) + Cov(\mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}], \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]) \\ & - & 2Cov(\boldsymbol{V_1}, \mathbb{E}[\boldsymbol{V_1} \mid \boldsymbol{V_2}]) \\ & & \\ & = & \Sigma_{11} + \Sigma_{12}\Sigma_{22}^{+}\Sigma_{21} - 2\Sigma_{12}\Sigma_{22}^{+}\Sigma_{21} \\ & = & \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{+}\Sigma_{21} \end{array}$$

■

9.2.3 Proof of proposition 3

This subsection deals with the proof of proposition 3, which gives the equivalence between all the ways of conditioning a Gaussian vector. As a recall, $(U, \boldsymbol{V}, \boldsymbol{W})$ is a Gaussian vector with U a Gaussian variable and $\boldsymbol{V}$ and $\boldsymbol{W}$ Gaussian vectors.

Proof

- *Case where all variables are centered.*
  - *First way of conditioning* U*:* $Z_1 = [U \mid \boldsymbol{V} = \boldsymbol{v}, \boldsymbol{W} = \boldsymbol{w}]$*.* U *can be decomposed as*

$$\begin{array}{rcl} U & = & \mathbb{E}[U \mid \boldsymbol{V}, \boldsymbol{W}] + T \\ & = & \alpha'\boldsymbol{V} + \beta'\boldsymbol{W} + T, \end{array} \tag{9.5}$$

    with $T \perp span(\boldsymbol{V}, \boldsymbol{W})$*. Using the decomposition* (9.5)*,* Z_1 *is equal to:*

$$Z_1 = \alpha'\boldsymbol{v} + \beta'\boldsymbol{w} + T.$$

  - *Second way of conditioning* U*:* $X = [U \mid \boldsymbol{V} = \boldsymbol{v}]$*,* $\boldsymbol{Y} = [\boldsymbol{W} \mid \boldsymbol{V} = \boldsymbol{v}]$ *and then* $Z_2 = [X \mid \boldsymbol{Y} = \boldsymbol{w}]$*. Using the decomposition of* U *in equation* (9.5)*,* X *is equal to:*

$$X = \alpha'\boldsymbol{v} + \beta'\boldsymbol{Y} + T$$

    because $T \perp \boldsymbol{V}$*. Then* Z_2 *is equal to*

$$Z_2 = \alpha'\boldsymbol{v} + \beta'\boldsymbol{w} + [T \mid \boldsymbol{Y} = \boldsymbol{w}].$$

    As $\boldsymbol{Y} \in span(\boldsymbol{V})$*,* $T \perp \boldsymbol{Y}$*, so* $[T \mid \boldsymbol{Y} = \boldsymbol{w}] = T$*. Finally:*

$$Z_2 = \alpha'\boldsymbol{v} + \beta'\boldsymbol{w} + T = Z_1.$$

- *General case: let* m_U, $\boldsymbol{m_V}$, *and* $\boldsymbol{m_W}$ *denote the means of* U, $\boldsymbol{V}$, *and* $\boldsymbol{W}$. *The previous formulae can be applied using* $\hat{U} = U - m_U$, $\hat{\boldsymbol{V}} = \boldsymbol{V} - \boldsymbol{m_V}$, $\hat{\boldsymbol{v}} = \boldsymbol{v} - \boldsymbol{m_V}$, $\hat{\boldsymbol{W}} = \boldsymbol{W} - \boldsymbol{m_W}$, $\hat{\boldsymbol{w}} = \boldsymbol{w} - \boldsymbol{m_W}$, $\hat{X} = \left[\hat{U} \mid \hat{\boldsymbol{V}} = \hat{\boldsymbol{v}}\right]$, $\hat{Y} = \left[\hat{\boldsymbol{W}} \mid \hat{\boldsymbol{V}} = \hat{\boldsymbol{v}}\right]$, $\hat{Z}_1 = \left[\hat{U} \mid \hat{\boldsymbol{V}} = \hat{\boldsymbol{v}}, \hat{\boldsymbol{W}} = \hat{\boldsymbol{w}}\right]$, *and* $\hat{Z}_2 = \left[\hat{X} \mid \hat{\boldsymbol{Y}} = \hat{\boldsymbol{w}}\right]$. $\hat{Z}_1$ *can be rewritten as*

$$\begin{aligned} \hat{Z}_1 &= \left[\hat{U} \mid \hat{\boldsymbol{V}} = \hat{\boldsymbol{v}}, \hat{\boldsymbol{W}} = \hat{\boldsymbol{w}}\right] \\ &= \left[U - m_U \mid \boldsymbol{V} - \boldsymbol{m_V} = \boldsymbol{v} - \boldsymbol{m_V}, \hat{\boldsymbol{W}} - \boldsymbol{m_W} = \boldsymbol{w} - \boldsymbol{m_W}\right] \\ &= [U \mid \boldsymbol{V} = \boldsymbol{v}, \boldsymbol{W} = \boldsymbol{w}] - m_U \\ &= Z_1 - m_U. \end{aligned}$$

$\hat{X}$ *can be rewritten as*

$$\begin{aligned} \hat{X} &= \left[\hat{U} \mid \hat{\boldsymbol{V}} = \hat{\boldsymbol{v}}\right] \\ &= [U \mid \boldsymbol{V} = \boldsymbol{v}] - m_U \\ &= X - m_U. \end{aligned}$$

Similarly, $\hat{\boldsymbol{Y}}$ *is equal to* $\hat{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{m_W}$. *Finally* $\hat{Z}_2$ *is equal to*

$$\begin{aligned} \hat{Z}_2 &= \left[\hat{X} \mid \hat{\boldsymbol{Y}} = \hat{\boldsymbol{w}}\right] \\ &= [X \mid \boldsymbol{Y} = \boldsymbol{w}] - m_U \\ &= Z_2 - m_U. \end{aligned}$$

The study of the previous case implies $\hat{Z}_1 = \hat{Z}_2$. *As a consequence* $Z_1 = Z_2$. ∎

9.2.4 Proof of proposition 4 [Friedman et al., 2001]

This subsection deals with the proof of proposition 4, which says that the likelihood increases at each iteration of the EM algorithm.

Proof *By definition of the conditional density*

$$\begin{aligned} h_{Z;\hat{\eta}}(z) &= \frac{h_{Z,Z^m;\hat{\eta}}(z,z^m)}{h_{Z^m|Z=z;\hat{\eta}}(z^m)} \\ &= \frac{h_{T;\hat{\eta}}(t)}{h_{Z^m|Z=z;\hat{\eta}}(z^m)} \end{aligned}$$

Taking the log*:*

$$l(\hat{\eta}, z) = l_0(\hat{\eta}, t) - \log(h_{Z^m|Z=z;\hat{\eta}}(z^m))$$

Randomizing the expression:

$$l(\hat{\eta}, Z) = l_0(\hat{\eta}, T) - \log(h_{Z^m|Z=z;\hat{\eta}}(Z^m)\mathbf{1}_{Z=z})$$

Taking the expectation conditionally to $Z = z$ *with the assumption that* Z *distribution is*

parameterized by $\hat{\eta}^{(k)}$*:*

$$\begin{aligned} l(\hat{\eta}; z) &= \underbrace{\mathbb{E}_{\hat{\eta}^{(k)}}\left[l_0(\hat{\eta}; T) \mid Z = z\right]}_{\mathcal{Q}(\hat{\eta}, \hat{\eta}^{(k)})} - \mathbb{E}_{\hat{\eta}^{(k)}}\left[\log\left(h_{Z^m|Z=z;\hat{\eta}}(Z^m)\right) \mid Z = z\right] \\ &= \mathcal{Q}(\hat{\eta}, \hat{\eta}^{(k)}) - \mathbb{E}_{\hat{\eta}^{(k)}}\left[\log\left(h_{\underbrace{Z^m \mid Z = z}_{\widetilde{Z}^m};\hat{\eta}}(\underbrace{Z^m \mid Z = z}_{\widetilde{Z}^m})\right)\right] \\ &= \mathcal{Q}(\hat{\eta}, \hat{\eta}^{(k)}) - \underbrace{\mathbb{E}_{\hat{\eta}^{(k)}}\left[\log\left(h_{\widetilde{Z}^m;\hat{\eta}}(\widetilde{Z}^m)\right)\right]}_{\mathcal{R}(\hat{\eta}, \hat{\eta}^{(k)})} \\ &= \mathcal{Q}(\hat{\eta}, \hat{\eta}^{(k)}) - \mathcal{R}(\hat{\eta}, \hat{\eta}^{(k)}) \end{aligned}$$

Then:

$$l(\hat{\eta}^{(k+1)}, z) - l(\hat{\eta}^{(k)}, z) = \left(\mathcal{Q}(\hat{\eta}^{(k+1)}, \hat{\eta}^{(k)}) - \mathcal{Q}(\hat{\eta}^{(k)}; \hat{\eta}^{(k)})\right) - \left(\mathcal{R}(\hat{\eta}^{(k+1)}, \hat{\eta}^{(k)}) - \mathcal{R}(\hat{\eta}^{(k)}; \hat{\eta}^{(k)})\right)$$

By definition of $\hat{\eta}^{(k+1)}$*:*

$$\mathcal{Q}(\hat{\eta}^{(k+1)}, \hat{\eta}^{(k)}) - \mathcal{Q}(\hat{\eta}^{(k)}; \hat{\eta}^{(k)}) \geqslant 0$$

So it must be shown that:

$$R = \mathcal{R}(\hat{\eta}^{(k+1)}, \hat{\eta}^{(k)}) - \mathcal{R}(\hat{\eta}^{(k)}; \hat{\eta}^{(k)}) \leqslant 0$$

Indeed:

$$
\begin{aligned}
R &= \mathbb{E}_{\hat{\eta}^{(k)}}\left[\log\left(h_{\tilde{Z}^m;\hat{\eta}^{(k+1)}}(\tilde{Z}^m)\right)\right] - \mathbb{E}_{\hat{\eta}^{(k)}}\left[\log\left(h_{\tilde{Z}^m;\hat{\eta}^{(k)}}(\tilde{Z}^m)\right)\right] \\
&= \mathbb{E}_{\hat{\eta}^{(k)}}\left[\log\left(\frac{h_{\tilde{Z}^m;\hat{\eta}^{(k+1)}}(\tilde{Z}^m)}{h_{\tilde{Z}^m;\hat{\eta}^{(k)}}(\tilde{Z}^m)}\right)\right] \\
&\leqslant \log\left(\mathbb{E}_{\hat{\eta}^{(k)}}\left[\underbrace{\frac{h_{\tilde{Z}^m;\hat{\eta}^{(k+1)}}(\tilde{Z}^m)}{h_{\tilde{Z}^m;\hat{\eta}^{(k)}}(\tilde{Z}^m)}}_{g(\tilde{Z}^m)}\right]\right) \\
&\quad \textit{(Jensen inequality applied to a concave function)} \\
&\leqslant \log\left(\mathbb{E}_{\hat{\eta}^{(k)}}\left[g(\tilde{Z}^m)\right]\right) \\
&\leqslant \log\left(\int g(\tilde{z}^m) h_{\tilde{Z}^m;\hat{\eta}^{(k)}}(\tilde{z}^m)\mathrm{d}\mu(\tilde{z}^m)\right) \\
&\quad \textit{Law of the unconscious statistician} \\
&\leqslant \log\left(\int \frac{h_{\tilde{Z}^m;\hat{\eta}^{(k+1)}}(\tilde{z}^m)}{h_{\tilde{Z}^m;\hat{\eta}^{(k)}}(\tilde{z}^m)} h_{\tilde{Z}^m;\hat{\eta}^{(k)}}(\tilde{z}^m)\mathrm{d}\mu(\tilde{z}^m)\right) \\
&\leqslant \log\left(\int h_{\tilde{Z}^m;\hat{\eta}^{(k+1)}}(\tilde{z}^m)\mathrm{d}\mu(\tilde{z}^m)\right) \\
&\leqslant \log(1) \\
&\leqslant 0
\end{aligned}
$$

■

9.2.5 Proof of proposition 5 [Zertuche, 2015]

This subsection deals with the proof of proposition 5 which gives the formulae used in the EM algorithm.

Proof • *The randomized complete data loglikelihood is equal to*

$$
\begin{aligned}
l_0(\eta; Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_2), Y_c(\mathbb{X}_2)) &= \log h_{Y_1(\mathbb{X}_1),Y_1(\mathbb{X}_2),Y_c(\mathbb{X}_2);\eta}\left(Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_2), Y_c(\mathbb{X}_2)\right) \\
&= \log h_{Y_1(\mathbb{X}_1),Y_1(\mathbb{X}_2);\eta_1}\left(Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_2)\right) + \log h_{Y_c(\mathbb{X}_2);\eta_c}\left(Y_c(\mathbb{X}_2)\right)
\end{aligned}
$$

• *The expectation to maximize is*

$$
\begin{aligned}
\mathcal{Q}(\eta, \eta^*) &= \mathbb{E}_{\eta^*}\left[l_0(\eta; Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_2), Y_c(\mathbb{X}_2)) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, Y_2(\mathbb{X}_2) = \boldsymbol{y_2}\right] \\
&= \underbrace{\mathbb{E}_{\eta^*}\left[\log h_{Y_1(\mathbb{X}_1),Y_1(\mathbb{X}_2);\eta_1}\left(Y_1(\mathbb{X}_1), Y_1(\mathbb{X}_2)\right) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, Y_2(\mathbb{X}_2) = \boldsymbol{y_2}\right]}_{=\mathcal{Q}_1(\eta_1,\eta^*)} \\
&\quad + \underbrace{\mathbb{E}_{\eta^*}\left[\log h_{Y_c(\mathbb{X}_2);\eta_c}\left(Y_c(\mathbb{X}_2)\right) \mid Y_1(\mathbb{X}_1) = \boldsymbol{y_1}, Y_2(\mathbb{X}_2) = \boldsymbol{y_2}\right]}_{\mathcal{Q}_c(\eta_c,\eta^*)}
\end{aligned}
$$

- $\mathcal{Q}_1$ *can be rewritten as:*

$$\begin{aligned}
\mathcal{Q}_1(\eta_1,\eta^*) &= \mathbb{E}_{\eta^*}\left[\log h_{Y_1(\mathbb{X}_1),Y_1(\mathbb{X}_2);\eta_1}\left(Y_1(\mathbb{X}_1),Y_1(\mathbb{X}_2)\right) \mid Y_1(\mathbb{X}_1)=\boldsymbol{y_1}, Y_2(\mathbb{X}_2)=\boldsymbol{y_2}\right] \\
&= \mathbb{E}_{\eta^*}\left[\log h_{Y_1(\mathbb{X}_1),Y_1(\mathbb{X}_2);\eta_1}\left(\boldsymbol{y_1}, \underbrace{Y_1(\mathbb{X}_2) \mid Y_1(\mathbb{X}_1)=\boldsymbol{y_1}, Y_2(\mathbb{X}_2)=\boldsymbol{y_2}}_{=\widetilde{Y}_1(\mathbb{X}_2)}\right)\right] \\
&= -\tfrac{n_1+n_2}{2}\log 2\pi - \tfrac{1}{2}\log|\Sigma_{\theta_1}| \\
&\quad - \mathbb{E}_{\eta^*}\left[\frac{\left((\boldsymbol{y_1}-m_1\mathbf{1}_{n_1})' \quad \left(\widetilde{Y}_1(\mathbb{X}_2)-m_1\mathbf{1}_{n_2}\right)'\right)\Sigma_{\theta_1}^{-1}\begin{pmatrix}\boldsymbol{y_1}-m_1\mathbf{1}_{n_1}\\ \widetilde{Y}_1(\mathbb{X}_2)-m_1\mathbf{1}_{n_2}\end{pmatrix}}{2}\right] \\
&= -\tfrac{n_1+n_2}{2}\log 2\pi - \tfrac{1}{2}\log|\Sigma_{\theta_1}| \\
&\quad \underbrace{-\tfrac{1}{2}\,tr\left(\Sigma_{\theta_1}^{-1} cov_{\eta^*}\left(\begin{pmatrix}\boldsymbol{y_1}-m_1\mathbf{1}_{n_1}\\ \widetilde{Y}_1(\mathbb{X}_2)-m_1\mathbf{1}_{n_2}\end{pmatrix},\begin{pmatrix}\boldsymbol{y_1}-m_1\mathbf{1}_{n_1}\\ \widetilde{Y}_1(\mathbb{X}_2)-m_1\mathbf{1}_{n_2}\end{pmatrix}\right)\right)}_{=t} \\
&\quad -\tfrac{1}{2}\left((\boldsymbol{y_1}-m_1\mathbf{1}_{n_1})' \quad \left(\underbrace{\mathbb{E}_{\eta^*}\left[\widetilde{Y}_1(\mathbb{X}_2)\right]}_{=\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta^*}}} - m_1\mathbf{1}_{n_2}\right)'\right)\Sigma_{\theta_1}^{-1}\left(\underbrace{\mathbb{E}_{\eta^*}\left[\widetilde{Y}_1(\mathbb{X}_2)\right]}_{=\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta^*}}} - m_1\mathbf{1}_{n_2}\right)
\end{aligned}$$

The last equality is obtained using the expectation of a quadratic form formula. Using the componentwise matrix inversion of Σ_{θ_1}, the trace term is equal to:

$$\begin{aligned}
t &= tr\left(\Sigma_{\theta_1}^{-1}\begin{pmatrix}0_{n_1,n_1} & 0_{n_1,n_2}\\ 0_{n_2,n_1} & \underbrace{cov_{\eta^*}(\widetilde{Y}_1(\mathbb{X}_2),\widetilde{Y}_1(\mathbb{X}_2))}_{=\widetilde{\Sigma}_{\eta^*}}\end{pmatrix}\right) \\
&= tr\left(\left(k_{\theta_1}(\mathbb{X}_2,\mathbb{X}_2)-k_{\theta_1}(\mathbb{X}_2,\mathbb{X}_1)k_{\theta_1}(\mathbb{X}_1,\mathbb{X}_1)^{-1}k_{\theta_1}(\mathbb{X}_1,\mathbb{X}_2)\right)^{-1}\widetilde{\Sigma}_{\eta^*}\right)
\end{aligned}$$

- $\mathcal{Q}_c$ *can be rewritten as*

$$\begin{aligned}
\mathcal{Q}_c(\eta_c,\eta^*) &= \mathbb{E}_{\eta^*}\left[\log h_{Y_c(\mathbb{X}_2);\eta_c}\left(Y_c(\mathbb{X}_2)\right) \mid Y_1(\mathbb{X}_1)=\boldsymbol{y_1}, Y_2(\mathbb{X}_2)=\boldsymbol{y_2}\right] \\
&= \mathbb{E}_{\eta^*}\left[\log h_{Y_c(\mathbb{X}_2);\eta_c}\left(Y_2(\mathbb{X}_2)-g(\mathbb{X}_2)\circ Y_1(\mathbb{X}_2)\right) \mid Y_1(\mathbb{X}_1)=\boldsymbol{y_1}, Y_2(\mathbb{X}_2)=\boldsymbol{y_2}\right] \\
&= \mathbb{E}_{\eta^*}\left[\log h_{Y_c(\mathbb{X}_2);\eta_c}\left(\boldsymbol{y_2}-g(\mathbb{X}_2)\circ\widetilde{Y}_1(\mathbb{X}_2)\right)\right] \\
&= -\tfrac{n_2}{2}\log 2\pi - \tfrac{1}{2}\log|\Sigma_{\theta_c}| \\
&\quad -\tfrac{1}{2}\mathbb{E}_{\eta^*}\left[\left(\boldsymbol{y_2}-g(\mathbb{X}_2)\circ\widetilde{Y}_1(\mathbb{X}_2)-m_c\mathbf{1}_{n_2}\right)'\Sigma_{\theta_c}^{-1}\left(\boldsymbol{y_2}-g(\mathbb{X}_2)\circ\widetilde{Y}_1(\mathbb{X}_2)-m_c\mathbf{1}_{n_2}\right)\right] \\
&= -\tfrac{n_2}{2}\log(2\pi) - \tfrac{1}{2}\log(|\Sigma_{\theta_d}|) \\
&\quad -\tfrac{1}{2}tr\left(\Sigma_{\theta_d}^{-1}diag(g(\mathbb{X}_2))\widetilde{\Sigma}_{\eta^*}diag(g(\mathbb{X}_2))\right) \\
&\quad -\tfrac{1}{2}\left(\mathbb{Y}_2-g(\mathbb{X}_2)\circ\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta^*}}-m_d\mathbf{1}_{n_2}\right)'\Sigma_{\theta_d}^{-1}\left(\mathbb{Y}_2-g(\mathbb{X}_2)\circ\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\eta^*}}-m_d\mathbf{1}_{n_2}\right)
\end{aligned}$$

■

9.2.6 Proof of proposition 6 [Le Gratiet, 2013a]

This subsection deals with the proof of proposition 6 which gives a formula of the sobol index easy to implement by a Monte-Carlo estimation.

Proof *Let* $X = (X_1, \cdots, X_{i-1}, X_i, X_{i+1}, \cdots, X_d)$ *be a random vector. Let* $\widetilde{X} = (\widetilde{X}_1, \cdots, \widetilde{X}_{i-1}, X_i, \widetilde{X}_{i+1}, \cdots, \widetilde{X}_d)$ *be another random vector with* $\widetilde{X}_j$ *independent of same law than* X_j*. The covariance between* $f(X)$ *and* $f(\widetilde{X})$ *can be rewritten as:*

$$\begin{aligned}
Cov(f(X), f(\widetilde{X})) &= \mathbb{E}\left[f(X)f(\widetilde{X})\right] - \mathbb{E}\left[f(X)\right]\mathbb{E}\left[f(\widetilde{X})\right] \\
&= \mathbb{E}\left[f(X)f(\widetilde{X})\right] - \mathbb{E}\left[f(X)\right]^2, \text{ because } f(X) \overset{\mathcal{L}}{=} f(\widetilde{X}) \\
&= \mathbb{E}\left[\mathbb{E}\left[f(X)f(\widetilde{X}) \mid X_i\right]\right] - \mathbb{E}\left[\mathbb{E}\left[f(X) \mid X_i\right]\right]^2 \\
&= \mathbb{E}\left[\mathbb{E}\left[f(X) \mid X_i\right]\mathbb{E}\left[f(\widetilde{X}) \mid X_i\right]\right] - \mathbb{E}\left[\mathbb{E}\left[f(X) \mid X_i\right]\right]^2, \text{ because } \widetilde{X}_j \perp X_j \\
&= \mathbb{E}\left[\mathbb{E}\left[f(X) \mid X_i\right]^2\right] - \mathbb{E}\left[\mathbb{E}\left[f(X) \mid X_i\right]\right]^2 \\
&= Var(\mathbb{E}\left[f(X) \mid X_i\right]).
\end{aligned}$$

■

9.2.7 Proof of proposition 7

This subsection deals with the proof of proposition 7 which gives a formula of the expectation of a centered Gaussian process $\widetilde{Z}$ of kernel k conditioned on any set of points $\mathcal{D}$ indexed by a set $\mathcal{S}$ of measure ν. This expectation is defined as the projection on the Gaussian space generated by $\{\widetilde{Z}(t), t \in \mathcal{D}\}$ and is denoted by:

$$P(\widetilde{Z})(x) = \mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(t), t \in \mathcal{D}\right].$$

The proof given in this appendix is different from the one proposed by [Gauthier, 2011]. It uses the Karhunen-Loeve decomposition.

Proof • *The first step consists in focusing on the restriction of* $P(\widetilde{Z})$ *on* $\mathcal{D}$*, as in this case* $P(\widetilde{Z}) = \widetilde{Z}$*. They can be considered as functions of* s *defined on* $\mathcal{S}$*. The Karhunen-Loeve decomposition of* $\widetilde{Z}$ *(and so of* $P(\widetilde{Z})$*) on* $\mathcal{D}$ *is:*

$$P(\widetilde{Z})(x_s) = \sum_{n=1}^{+\infty} \widetilde{\phi}_n(s) \int_{\mathcal{S}} \widetilde{\phi}_n(s)\widetilde{Z}(x_s)\mathrm{d}\nu(s).$$

$(\widetilde{\phi}_n)_{n=1}^{+\infty}$ *is an orthonormal basis of* $L^2(S, \nu)$ *of eigenfunctions of the eigenvalue problem:*

$$\int_{\mathcal{S}} k(x_s, x_u)\widetilde{\phi}_n(u)\mathrm{d}\nu(u) = \lambda_n \widetilde{\phi}_n(s)$$

with

$$\int_{\mathcal{S}} \widetilde{\phi}_n(s)\widetilde{\phi}_m(s)\mathrm{d}\nu(s) = \delta_{nm}.$$

- $\widetilde{\phi}_n$ *can be seen as the restriction on* $\mathcal{D}$ *of a function* ϕ_n *defined on* $\mathcal{X}_Z$*:* $\widetilde{\phi}_n(s) = \phi_n(x_s)$*. Then, the expectation is searched of the form:*

$$P(\widetilde{Z})(x) = \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}(x_s) \mathrm{d}\nu(s), \ \forall x \in \mathcal{X}_Z.$$

The goal is to find ϕ_n *such that* $P(\widetilde{Z})(x)$ *is the orthogonal projection of* $\widetilde{Z}(x)$ *on the Gaussian space generated by* $\widetilde{Z}(\mathcal{D})$*:*

$$\begin{array}{rrcl} & Cov\left(\widetilde{Z}(x) - P(\widetilde{Z})(x), P(\widetilde{Z})(x)\right) & = & 0 \\ \Leftrightarrow & Cov\left(\widetilde{Z}(x), P(\widetilde{Z})(x)\right) - Cov\left(P(\widetilde{Z})(x), P(\widetilde{Z})(x)\right) & = & 0 \end{array} \tag{9.6}$$

The first term is equal to:

$$\begin{array}{rcl} Cov\left(\widetilde{Z}(x), P(\widetilde{Z})(x)\right) & = & Cov\left(\widetilde{Z}(x), \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}(x_s) \mathrm{d}\nu(s)\right) \\ & = & \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) Cov\left(\widetilde{Z}(x), \widetilde{Z}(x_s)\right) \mathrm{d}\nu(s) \\ & = & \sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) k(x, x_s) \mathrm{d}\nu(s). \end{array}$$

the second term is equal to:

$$\begin{aligned} & Cov\left(P(\widetilde{Z})(x), P(\widetilde{Z})(x)\right) = Cov\left(\sum_{n=1}^{+\infty} \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{Z}(x_s) \mathrm{d}\nu(s), \right. \\ & \left. \sum_{m=1}^{+\infty} \phi_m(x) \int_{\mathcal{S}} \widetilde{\phi}_m(u) \widetilde{Z}(x_u) \mathrm{d}\nu(u)\right) \\ & = \sum_{n=1}^{+\infty} \sum_{m=1}^{+\infty} \phi_n(x) \phi_m(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \int_{\mathcal{S}} \widetilde{\phi}_m(u) Cov\left(Z(x_s), Z(x_u)\right) \mathrm{d}\nu(u) \mathrm{d}\nu(s) \\ & = \sum_{n=1}^{+\infty} \sum_{m=1}^{+\infty} \phi_n(x) \phi_m(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s) \underbrace{\int_{\mathcal{S}} \widetilde{\phi}_m(u) k(x_s, x_u) \mathrm{d}\nu(u)}_{\lambda_m \widetilde{\phi}_m(s)} \mathrm{d}\nu(s) \\ & = \sum_{n=1}^{+\infty} \sum_{m=1}^{+\infty} \lambda_m \phi_n(x) \phi_m(x) \underbrace{\int_{\mathcal{S}} \widetilde{\phi}_n(s) \widetilde{\phi}_m(s) \mathrm{d}\nu(s)}_{\delta_{nm}} \\ & = \sum_{n=1}^{+\infty} \lambda_n \phi_n(x)^2. \end{aligned}$$

Using the expressions obtained above, equation (9.6) *becomes:*

$$\sum_{n=1}^{+\infty} \phi_n(x) \left[\int_{\mathcal{S}} \widetilde{\phi}_n(s) k(x, x_s) \mathrm{d}\nu(s) - \lambda_n \phi_n(x)\right] = 0.$$

This equation is satisfied by defining:

$$\phi_n(x) = \frac{1}{\lambda_n} \int_{\mathcal{S}} k(x, x_s) \widetilde{\phi}_n(s) \mathrm{d}\nu(s).$$

With this definition, $\widetilde{\phi}_n$ is the restriction of ϕ_n on $\mathcal{D}$:

$$\begin{aligned}\phi_n(x_s) &= \tfrac{1}{\lambda_n}\underbrace{\int_{\mathcal{S}} k(x_s,x_u)\widetilde{\phi}_n(u)\mathrm{d}\nu(u)}_{\lambda_n\widetilde{\phi}_n(s)} \\ &= \widetilde{\phi}_n(s).\end{aligned}$$

■

9.2.8 Proof of proposition 8 [Gauthier and Bay, 2012a]

This subsection recalls the proof of proposition 8, which establishes, when $\mathcal{D}=\mathbb{D}$ is finite, the equality between the two formulae of $Z(x)=\widetilde{Z}(x)-\mathbb{E}\left[\widetilde{Z}(x)\mid Z(\mathcal{D})\right]$, with $\widetilde{Z}$ a centered Gaussian process defined on $\mathcal{X}=\mathbb{R}^d$ of covariance kernel k, and $\mathcal{D}=\{x_s, s\in\mathcal{S}\}$ a subset of $\mathcal{X}$ indexed by $\mathcal{S}$ of measure ν. The first formula of this process is described in proposition 2:

$$Z(x)=\left[\widetilde{Z}(x)\mid \widetilde{Z}(\mathbb{D})=0\right]=\widetilde{Z}(x)-k(x,\mathbb{D})k(\mathbb{D},\mathbb{D})^{-1}\widetilde{Z}(\mathbb{D}).$$

The second formula is given in proposition 7, the reader can refer to it for the notations:

$$Z(x)=\widetilde{Z}(x)-\sum_{n=1}^{L}\phi_n(x)\int_{\mathcal{S}}\widetilde{\phi}_n(s)\widetilde{Z}(x_s)\mathrm{d}\nu(s).$$

Proof *Let $\mathbb{D}=\left\{x^{(1)},\ldots,x^{(L)}\right\}$ denote the elements of $\mathbb{D}$. $\mathbb{D}$ is indexed by $\mathcal{S}=\{1,\ldots,L\}$ of measure $\nu=\sum_{i=1}^{L}\omega_i\delta_i$, with δ_i the Dirac measure of integer i and $\omega_i>0$ the weight associated with i. $\mathcal{X}=\mathbb{R}^d$ is provided with the scalar product $(x,y)_W=x'Wy$ with $W=\begin{pmatrix}\omega_1 & 0 & \ldots & 0\\ 0 & \ddots & \ddots & \vdots\\ \vdots & \ddots & \ddots & 0\\ 0 & \ldots & 0 & \omega_L\end{pmatrix}$.*

- ***Eigenvalue problem.*** *The eigenvalue problem is equal to:*

$$\begin{aligned} & \textstyle\int_{\mathcal{S}} k(x_t,x_s)\widetilde{\phi}(s)\mathrm{d}\nu(s) &=& \lambda\widetilde{\phi}(t), \quad \forall t\in\mathcal{S},\\ \Leftrightarrow\ & \textstyle\sum_{j=1}^{L} k(x^{(i)},x^{(j)})\omega_j\widetilde{\phi}(j) &=& \lambda\widetilde{\phi}(i), \quad \forall i\in\{1,\ldots,L\},\\ \Leftrightarrow\ & k(\mathbb{D},\mathbb{D})W\widetilde{\phi}(\mathcal{S}) &=& \lambda\widetilde{\phi}(\mathcal{S}),\\ \Leftrightarrow\ & TW\widetilde{\phi}(\mathcal{S}) &=& \lambda\widetilde{\phi}(\mathcal{S}),\end{aligned}$$

  where $T=k(\mathbb{D},\mathbb{D})$, λ is an eigen value of the problem, and especially of the matrix TW, and $\widetilde{\phi}:\mathcal{S}\to\mathbb{R}$ is an eigen function of the problem and can be seen as an eigen vector of the matrix TW, by denoting it as $\widetilde{\phi}(\mathcal{S})$.

- ***Eigen values and orthonormal eigen vectors.*** *The endomorphism TW is a self-adoint endomorphism of $\mathbb{R}^d$ for the scalar product $(,)_W$:*

$$\begin{aligned}(TWx,y)_W &= (TWx)'Wy\\ &= x'WTWy\\ &= x'W(TWy)\\ &= (x,TWy)_W.\end{aligned}$$

As a consequence (see theorem 3 of [Gourdon, 1994]), there exist eigen vectors $(\widetilde{\phi}_n(\mathcal{S}))_{n=1}^L$ *of* TW *that form an orthonormal basis of* $\mathcal{X} = \mathbb{R}^d$ *for the scalar product* $(,)_W$*. Let denote by* $(\lambda_n)_{n=1}^L$ *the associated eigen values of* TW*, the spectral decomposition of* TW *is* $TW = P\Lambda P^{-1}$ *where*

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda_L \end{pmatrix}, \quad P = \begin{pmatrix} | & & | \\ \widetilde{\phi}_1(\mathcal{S}) & \dots & \widetilde{\phi}_L(\mathcal{S}) \\ | & & | \end{pmatrix},$$

with $P'WP = I_L$*.*

- ***Formula of*** ϕ_n***:*** *The function* $\phi_n(x)$ $(x \in \mathcal{X})$ *is equal to:*

$$\begin{aligned} \phi_n(x) &= \tfrac{1}{\lambda_n} \textstyle\int_{\mathcal{S}} k(x, x_s)\widetilde{\phi}_n(s)\mathrm{d}\nu(s) \\ &= \tfrac{1}{\lambda_n} k(x, \mathbb{D})W\widetilde{\phi}_n(\mathcal{S}) \\ &= \tfrac{1}{\lambda_n}\widetilde{\phi}_n(\mathcal{S})'Wk(\mathbb{D}, x). \end{aligned}$$

The functional vector $\phi(x) = \begin{pmatrix} \phi_1(x) \\ \vdots \\ \phi_L(x) \end{pmatrix}$ *is equal to:*

$$\phi(x) = \Lambda^{-1}P'Wk(\mathbb{D}, x).$$

- ***Value of the expectation****. The formula of the expectation* $\mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathbb{D})\right]$ *given in proposition 7 becomes:*

$$\begin{aligned} \mathbb{E}\left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathbb{D})\right] &= \textstyle\sum_{n=1}^L \phi_n(x) \int_{\mathcal{S}} \widetilde{\phi}_n(s)\widetilde{Z}(x_s)\mathrm{d}\nu(s) \\ &= \begin{pmatrix} \phi_1(x) & \dots & \phi_L(x) \end{pmatrix} \begin{pmatrix} \int_{\mathcal{S}} \widetilde{\phi}_1(s)\widetilde{Z}(x_s)\mathrm{d}\nu(s) \\ \vdots \\ \int_{\mathcal{S}} \widetilde{\phi}_L(s)\widetilde{Z}(x_s)\mathrm{d}\nu(s) \end{pmatrix} \\ &= \phi(x)' \begin{pmatrix} \widetilde{\phi}_1(\mathcal{S})'W\widetilde{Z}(\mathbb{D}) \\ \vdots \\ \widetilde{\phi}_L(\mathcal{S})'W\widetilde{Z}(\mathbb{D}) \end{pmatrix} \\ &= \phi(x)' \begin{pmatrix} \widetilde{\phi}_1(\mathcal{S})' \\ \vdots \\ \widetilde{\phi}_L(\mathcal{S})' \end{pmatrix} W\widetilde{Z}(\mathbb{D}) \\ &= \phi(x)'P'W\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})WP\Lambda^{-1}P'W\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})WP\Lambda^{-1}P'WPP^{-1}\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})WP\Lambda^{-1}P^{-1}\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})W(TW)^{-1}\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})WW^{-1}T^{-1}\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})T^{-1}\widetilde{Z}(\mathbb{D}) \\ &= k(x, \mathbb{D})k(\mathbb{D}, \mathbb{D})^{-1}\widetilde{Z}(\mathbb{D}). \end{aligned}$$

This formula coincides with the formula given in proposition 1 and gives the equality wanted. ∎

9.3 Proofs of PhD propositions

This section deals with all the proofs related to the analytical formulae of the P process in the context of the seqGPR metamodel.

9.3.1 Proof of proposition 9 for a Monte-Carlo method

This subsection deals with the proof of proposition 9, which establishes the equivalence between approximating the process Z^P (defined in equation (3.5)) by discretizing its spectral decomposition with a Monte-Carlo method and conditioning it on the discretization points.

Proof *$\mathcal{D} = \{(s_J, g(s_J)), s_J \in [0,1]^{d_J}\}$ is indexed by $\mathcal{S} = [0,1]^{d_J}$ of measure $\nu = \lambda$ (Lebesgues measure).*

- *The eigenvalue problem is equal to:*

$$\begin{array}{rll} & \int_{\mathcal{S}} k(x_{s_J}, x_{u_J}) \widetilde{\phi}(u_J) \mathrm{d}u_J & = \lambda \widetilde{\phi}(s_J), \quad \forall s_J \in \mathcal{S}, \\ \Leftrightarrow & \int_{[0,1]^{d_J}} \sigma^2 r((s_J, g(s_J)), (u_J, g(u_J))) \widetilde{\phi}(u_J) \mathrm{d}u_J & = \lambda \widetilde{\phi}(s_J) \quad \forall s_J \in [0,1]^{d_J}. \end{array}$$

It is discretized using a Monte-Carlo method. A sample $\mathbb{S} = (s_J^{(i)})_{1 \leqslant i \leqslant L}$ is generated uniformly in $[0,1]^{d_J}$ to build $\mathbb{D} = \left(s_J^{(i)}, g(s_J^{(i)})\right)_{1 \leqslant i \leqslant L}$ which is a discretization of the subspace $\mathcal{D} = \{(s_J, g(s_J)), s_J \in [0,1]^{d_J}\}$. The Monte-Carlo approximation of the integral in the eigenvalue problem is:

$$\frac{1}{L} \sum_{i=1}^{L} \sigma^2 r((s_J, g(s_J)), (s_J^{(i)}, g(s_J^{(i)}))) \widetilde{\phi}(s_J^{(i)}) \approx \lambda \widetilde{\phi}(s_J).$$

Considering only the values of $\widetilde{\phi}$ in $\mathbb{S}$, the eigenvalue problem becomes a finite dimensional one:

$$\left(\frac{1}{L} \sigma^2 r(\mathbb{D}, \mathbb{D})\right) \widetilde{\Phi} = \gamma \widetilde{\Phi}. \tag{9.7}$$

The solutions of (9.7) are noted $(\gamma_n, V_n)_{1 \leqslant n \leqslant L}$. The V_n are taken such that they verify:

$$V_n' \, V_m = \delta_{nm}.$$

The discretization of $\widetilde{\phi}_n$ (noted $\widetilde{\Phi}_n$) must verify the discrete equivalent of

$$\int_{[0,1]^{d_J}} \widetilde{\phi}_n(s_J) \widetilde{\phi}_m(s_J) \mathrm{d}s_J = \delta_{nm},$$

which is

$$\frac{1}{L} \, \widetilde{\Phi}_n' \, \widetilde{\Phi}_m = \delta_{nm}.$$

So the following relation between V_n and $\widetilde{\Phi}_n$ is verified: $V_n = \frac{1}{\sqrt{L}} \widetilde{\Phi}_n \Leftrightarrow \widetilde{\Phi}_n = \sqrt{L} V_n$.

- *Approximation of ϕ_n:*

  As

  $$\phi_n(x_J, x_I) \;=\; \tfrac{1}{\lambda_n}\int_{[0,1]^{d_J}} \sigma^2 r\left((x_J,x_I),(s_J,g(s_J))\right)\widetilde{\phi}_n(s_J)\mathrm{d}s_J \quad \forall (x_J,x_I)\in[0,1]^{d_J}\times[0,1]^{d_I},$$

  using the same Monte-Carlo approximation and replacing $(\lambda_n, \widetilde{\phi}_n)$ by $(\gamma_n, \widetilde{\Phi}_n)$, the following approximation of ϕ_n is obtained:

  $$\phi_n^{\mathbb{D}}(x) \;=\; \tfrac{1}{\gamma_n}\tfrac{1}{L}\sigma^2 r(x,\mathbb{D})\widetilde{\Phi}_n \quad \forall x\in[0,1]^{d_J+d_I}.$$

- *Decomposition of the matrix $\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1}$*

  $(\gamma_n, V_n)_{n\geqslant 1}$ are the eigenvalues and (orthonormal) eigenvectors of $\frac{1}{L}\sigma^2 r(\mathbb{D},\mathbb{D})$. So $(\frac{1}{\gamma_n}, V_n)_{n\geqslant 1}$ are the eigenvalues and (orthonormal) eigenvectors of $L\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1}$, and:

  $$\begin{aligned} L\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1} &= \sum_{n=1}^{L}\frac{1}{\gamma_n}V_nV_n', \\ &= \sum_{n=1}^{L}\frac{1}{\gamma_n}\left(\frac{1}{\sqrt{L}}\widetilde{\Phi}_n\right)\left(\frac{1}{\sqrt{L}}\widetilde{\Phi}_n'\right), \\ &= \frac{1}{L}\sum_{n=1}^{L}\gamma_n\widetilde{\Phi}_n\widetilde{\Phi}_n'. \end{aligned}$$

  So

  $$\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1} = \frac{1}{L^2}\sum_{n=1}^{L}\gamma_n\widetilde{\Phi}_n\widetilde{\Phi}_n'.$$

- *Approximation of the process Z^P:*

  The integral $\int_{[0,1]^{d_J}}\widetilde{\phi}_n(t_J)\widetilde{Z}(t_J,g(t_J))\mathrm{d}t_J$, involved in the formula of Z^P (see (3.5)), is discretized the same way as before. It becomes:

  $$\frac{1}{L}\widetilde{\Phi}_n'\widetilde{Z}(\mathbb{D}).$$

  The approximation of the process is:

  $$\begin{aligned} Z^{\mathbb{D}}(x) &= \widetilde{Z}(x) - \sum_{n=1}^{L}\phi_n^{\mathbb{D}}(x)\left(\frac{1}{L}\widetilde{\Phi}_n'\widetilde{Z}(\mathbb{D})\right), \\ &= \widetilde{Z}(x) - \sum_{n=1}^{L}\left(\frac{1}{L}\frac{1}{\gamma_n}\sigma^2 r(x,\mathbb{D})\widetilde{\Phi}_n\right)\left(\frac{1}{L}\widetilde{\Phi}_n'\widetilde{Z}(\mathbb{D})\right), \\ &= \widetilde{Z}(x) - \left(\sigma^2 r(x,\mathbb{D})\right)\left(\frac{1}{L^2}\sum_{n=1}^{L}\frac{1}{\gamma_n}\widetilde{\Phi}_n\widetilde{\Phi}_n'\right)\widetilde{Z}(\mathbb{D}), \\ &= \widetilde{Z}(x) - \left(\sigma^2 r(x,\mathbb{D})\right)\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1}\widetilde{Z}(\mathbb{D}), \\ &= \widetilde{Z}(x) - \mathbb{E}[\widetilde{Z}(x)\mid\widetilde{Z}(\mathbb{D})], \\ &= \left[\widetilde{Z}(x)\mid\widetilde{Z}(\mathbb{D})=0\right]. \end{aligned}$$

The process approximating Z^P is the conditioned (on a finite set of points) Gaussian process $Z^{\mathbb{D}}$. It is a centered Gaussian process of covariance kernel $\sigma^2\rho^{\mathbb{D}}$ with:

$$\rho^{\mathbb{D}}(x,x') \;=\; r(x,x') - r(x,\mathbb{D})r(\mathbb{D},\mathbb{D})^{-1}r(\mathbb{D},x') \quad \forall x,x'\in[0,1]^{d_J+d_I}.$$ ∎

9.3.2 Proof of proposition 10

This subsection deals with the proof of proposition 10 which gives a closed form formula of the process $Z^P = \widetilde{Z} - \mathbb{E}\left[\widetilde{Z} \mid \widetilde{Z}(\mathcal{D})\right]$ (where $\mathcal{D} = \{\underbrace{(s_J, g(s_J))}_{x_{s_J}}, s_J \in \mathcal{S}\}$ is indexed by $\mathcal{S} = [0,1]^{d_J}$ of measure ν) for a particular choice of the kernel of $\widetilde{Z}$: $k((x_J, x_I), (t_J, t_I)) = \sigma^2 r_J(x_J, t_J) r_I(x_I - g(x_J), t_I - g(t_J))$ with r_J and r_I two stationary correlation kernels. In this case, the formula is:

$$Z^P(x_J, x_I) = \widetilde{Z}(x_J, x_I) - r_I(x_I - g(x_J), 0)\widetilde{Z}(x_J, x_I).$$

Proof • *The eigenvalue problem is equal to:*

$$\begin{aligned}
& \textstyle\int_{\mathcal{S}} \left(\sigma^2 r\left(x_{s_J}, x_{u_J}\right)\right) \widetilde{\phi}_n(u_J)\mathrm{d}\nu(u_J) = \lambda_n \widetilde{\phi}_n(s_J),\ \forall s_J \in \mathcal{S}, \\
\Leftrightarrow\ & \textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r\left((s_J, g(s_J)), (u_J, g(u_J))\right)\right) \widetilde{\phi}_n(u_J)\mathrm{d}\nu(u_J) = \lambda_n \widetilde{\phi}_n(s_J),\ \forall s_J \in [0,1]^{d_J}, \\
\Leftrightarrow\ & \textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r_J(s_J, u_J)\right) r_I\left(g(s_J) - g(s_J), g(u_J) - g(u_J)\right) \widetilde{\phi}_n(u_J)\mathrm{d}\nu(u_J) = \lambda_n \widetilde{\phi}_n(s_J), \\
\Leftrightarrow\ & \textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r_J(s_J, u_J)\right) \underbrace{r_I(0,0)}_{=1} \widetilde{\phi}_n(u_J)\mathrm{d}\nu(u_J) = \lambda_n \widetilde{\phi}_n(s_J), \\
\Leftrightarrow\ & \textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r_J(s_J, u_J)\right) \widetilde{\phi}_n(u_J)\mathrm{d}\nu(u_J) = \lambda_n \widetilde{\phi}_n(s_J).
\end{aligned} \tag{9.8}$$

- *Expression of ϕ_n:*
  ϕ_n *can be rewritten as (for* (x_J, x_I) *in* $[0,1]^{d_J} \times [0,1]^{d_I}$*):*

$$\begin{aligned}
\phi_n(x_J, x_I) &= \tfrac{1}{\lambda_n} \textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r\left((x_J, x_I), (s_J, g(s_J))\right)\right) \widetilde{\phi}_n(s_J)\mathrm{d}\nu(s_J), \\
&= \tfrac{1}{\lambda_n} \textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) r_I\left(x_I - g(x_J), g(s_J) - g(s_J)\right) \widetilde{\phi}_n(s_J)\mathrm{d}\nu(s_J), \\
&= \tfrac{1}{\lambda_n} \left(\textstyle\int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) \widetilde{\phi}_n(s_J)\mathrm{d}\nu(s_J)\right) r_I\left(x_I - g(x_J), 0\right)), \\
&= \tfrac{1}{\lambda_n} \left(\lambda_n \widetilde{\phi}_n(x_J)\right) r_I\left(x_I - g(x_J), 0\right)), \\
&= \widetilde{\phi}_n(x_J) r_I\left(x_I - g(x_J), 0\right)).
\end{aligned}$$

  The second last equality is due to the fact that $\widetilde{\phi}_n$ *is solution of the eigenvalue problem* (9.8)*.*

- *The P process can be rewritten as (for* (x_J, x_I) *in* $[0,1]^{d_J} \times [0,1]^{d_I}$*):*

$$\begin{aligned}
Z^P(x_J, x_I) &= \widetilde{Z}(x_J, x_I) - \sum_{n=1}^{+\infty} \phi_n(x_J, x_I) \int_{[0,1]^{d_J}} \widetilde{\phi}_n(s_J)\widetilde{Z}(s_J, g(s_J))\mathrm{d}\nu(s_J), \\
&= \widetilde{Z}(x_J, x_I) - \sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) r_I\left(x_I - g(x_J), 0\right)) \int_{[0,1]^{d_J}} \widetilde{\phi}_n(s_J)\widetilde{Z}(s_J, g(s_J))\mathrm{d}\nu(s_J), \\
&= \widetilde{Z}(x_J, x_I) - \left(\sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) \int_{[0,1]^{d_J}} \widetilde{\phi}_n(s_J)\widetilde{Z}(s_J, g(s_J))\mathrm{d}\nu(s_J)\right) r_I\left(x_I - g(x_J), 0\right),
\end{aligned}$$

  and, because $\widetilde{Z}(x_J, g(x_J))$ *belongs to the sub Gaussian space engendered by the* $\{\widetilde{Z}(s_J, g(s_J))\ \forall s_J \in [0,1]^{d_J}\}$*, its projection in this subspace is equal to itself:*

$$\begin{aligned}
\widetilde{Z}(x_J, g(x_J)) &= \mathbb{E}[\widetilde{Z}(x_J, g(x_J)) \mid \widetilde{Z}(s_J, g(s_J))\forall s_J],\ \forall x_J \in [0,1]^{d_J},\\
&= \sum_{n=1}^{+\infty} \phi_n(x_J, g(x_J)) \int \widetilde{\phi}_n(s_J)\widetilde{Z}(s_J, g(s_J))\mathrm{d}\nu(s_J),\\
&= \sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) \underbrace{r_I\left(g(x_J) - g(x_J), 0\right)}_{=1} \int \widetilde{\phi}_n(s_J)\widetilde{Z}(s_J, g(s_J))\mathrm{d}\nu(s_J),\\
&= \sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) \int \widetilde{\phi}_n(s_J)\widetilde{Z}(s_J, g(s_J))\mathrm{d}\nu(s_J).
\end{aligned}$$

The second equality is the formula of the conditional expectation (see (3.6)*).*

Finally:

$$Z^P(x) = \widetilde{Z}(x) - r_I\left(x_I - g(x_J), 0\right)\widetilde{Z}(x_J, g(x_J)).$$ ∎

9.3.3 Proof of proposition 13

This subsection deals with the proof of proposition 13, which gives an analytical formula of the P process in the case where the conditioning is done on multiple subspaces.

Proof *The notations of subsection 2.2.2 in chapter 2 are adapted to proposition 13:* $\mathcal{D} = \{x_s, s \in \mathcal{S}\}$*, with* $\mathcal{S} = [0,1]^{d_J} \times \mathbb{B}$*.* $\nu = \lambda \otimes (\sum_{i=1}^{n_{\mathbb{B}}} \delta_{b_i})$ *is choosen as a measure on* $\mathcal{S}$*.* λ *is the Lebesgues measure on* $[0,1]^{d_J}$*,* b_i *is the* i*th component of* $\mathbb{B}$*,* K *is the size of* $\mathbb{B}$*, and* δ_{b_i} *is the Dirac measure at* b_i*.*

- *The eigenvalue problem can be rewritten as (for* (s_I, s_J) *in* $\mathcal{S}$*):*

$$\begin{aligned}
& \int_{\mathcal{S}} \sigma^2 r((s_J, g(s_J) + s_I - g(s_J)), (t_J, g(t_J) + t_I - g(t_J)))\widetilde{\phi}_n(t_J, t_I)\mathrm{d}\nu(t_J, t_I) &=& \lambda_n\widetilde{\phi}_n(s_J, s_I)\\
\leftrightarrow\ & \int_{\mathcal{S}} \sigma^2 r((s_J, s_I), (t_J, t_I))\widetilde{\phi}_n(t_J, t_I)\mathrm{d}\nu(t_J, t_I) &=& \lambda_n\widetilde{\phi}_n(s_J, s_I)\\
\Leftrightarrow\ & \int_{[0,1]^{d_J}} \sigma^2 r_J(s_J, t_J)\left(\int_{\mathcal{B}} r_I(s_I, t_I)\widetilde{\phi}_n(t_J, t_I)(\sum_{i=1}^{K} \mathrm{d}\delta_{b_i}(t_I))\right)\mathrm{d}t_J &=& \lambda_n\widetilde{\phi}_n(s_J, s_I),\\
\Leftrightarrow\ & \int_{[0,1]^{d_J}} \sigma^2 r_J(s_J, t_J)\left(\sum_{i=1}^{K} r_I(s_I, b_i)\widetilde{\phi}_n(t_J, b_i)\right)\mathrm{d}t_J &=& \lambda_n\widetilde{\phi}_n(s_J, s_I),
\end{aligned}$$

  This eigenvalue problem has the following matricial form:

$$\sigma^2 r_I(\mathcal{B}, \mathcal{B})\left(\int_{[0,1]^{d_J}} r_J(s_J, t_J)\widetilde{\phi}_n(t_J, \mathcal{B})\mathrm{d}t_J\right) = \lambda_n\widetilde{\phi}_n(s_J, \mathcal{B}) \quad \forall s_J \in [0,1]^{d_J}, \tag{9.9}$$

  with $r_I(\mathcal{B}, \mathcal{B}) = (r_I(b_i, b_j))_{1 \leqslant i \leqslant j \leqslant K}$ *the covariance matrix composed of the* $r_I(b_i, b_j)$*, and* $\widetilde{\phi}_n(\cdot, \mathcal{B})$ *the column vector whose components are the functions* $x_J \mapsto \widetilde{\phi}_n(x_J, b_i)$ *.*

- $\phi_n(x_J, x_I)$ *(*$n \geqslant 1$*,* $(x_J, x_I) \in [0,1]^{d_J + d_I}$*) is equal to:*

$$\begin{aligned}
\phi_n(x_J, x_I) &= \tfrac{1}{\lambda_n} \int_{[0,1]^{d_J} \times \mathcal{B}} \sigma^2 r((x_J, x_I), (t_J, t_I))\widetilde{\phi}_n(t_J, t_I)\mathrm{d}\nu(t_J, t_I),\\
&= \tfrac{\sigma^2}{\lambda_n} \int_{[0,1]^{d_J}} r_J(x_J, t_J)\left(\sum_{t_I \in \mathcal{B}} r_I(x_I, t_I)\widetilde{\phi}_n(t_J, t_I)\mathrm{d}\nu(t_I)\right)dt_J\\
&= r_I(x_I, \mathcal{B})\left(\tfrac{\sigma^2}{\lambda_n} \int_{[0,1]^{d_J}} r_J(x_J, t_J)\widetilde{\phi}_n(t_J, \mathcal{B})\mathrm{d}t_J\right)\\
&= r_I(x_I, \mathcal{B}) r_I(\mathcal{B}, \mathcal{B})^{-1}\left(\tfrac{\sigma^2 r_I(\mathcal{B}, \mathcal{B})}{\lambda_n} \int_{[0,1]^{d_J}} r_J(x_J, t_J)\widetilde{\phi}_n(t_J, \mathcal{B})\mathrm{d}t_J\right)\\
&= r_I(x_I, \mathcal{B}) r_I(\mathcal{B}, \mathcal{B})^{-1}\widetilde{\phi}_n(x_J, \mathcal{B}) \text{ (using (9.9)).}
\end{aligned}$$

- *The expectation* $\mathbb{E}\left[\widetilde{Z}(x_J,x_I) \mid \widetilde{Z}(t_J,g(t_J)+t_I)\ \forall (t_J,t_I)\in[0,1]^{d_J}\times\mathcal{B}\right]$ *can be rewritten (for* (x_J,x_I) *in* $[0,1]^{d_J+d_I}$*):*

$$\begin{aligned}
&\mathbb{E}\left[\widetilde{Z}(x_J,x_I) \mid \widetilde{Z}(t_J,g(t_J)+t_I)\ \forall (t_J,t_I)\in[0,1]^{d_J}\times\mathcal{B}\right]\\
&=\sum_{n=1}^{+\infty}\phi_n(x_J,x_I)\int_{[0,1]^{d_J}\times\mathcal{B}}\widetilde{\phi}_n(t_J,t_I)\widetilde{Z}(t_J,t_I)\mathrm{d}\mu(t_J,t_I)\\
&=r_I(x_I,\mathcal{B})r_I(\mathcal{B},\mathcal{B})^{-1}\left(\sum_{n=1}^{+\infty}\widetilde{\phi}_n(x_J,\mathcal{B})\int_{[0,1]^{d_J}\times\mathbb{B}}\widetilde{\phi}_n(t_J,t_I)\widetilde{Z}(t_J,t_I)\mathrm{d}\mu(t_J,t_I)\right)
\end{aligned}$$

and, because $\widetilde{Z}(x_J,g(x_J)+\mathcal{B})$ *belongs to the sub Gaussian space engendered by the* $\{\widetilde{Z}(t_J,g(t_J)+t_I)\ \forall (t_J,t_I)\in[0,1]^{d_J}\times\mathcal{B}\}$*, its projection in this subspace is itself:*

$$\begin{aligned}
\widetilde{Z}(x_J,g(x_J)+\mathcal{B}) &= \mathbb{E}\left[\widetilde{Z}(x_J,g(x_J)+\mathcal{B}) \mid \widetilde{Z}(t_J,g(t_J)+t_I)\ \forall (t_J,t_I)\in[0,1]^{d_J}\times\mathcal{B}\right]\\
&= \sum_{n=1}^{+\infty}\phi_n(x_J,\mathcal{B})\int_{[0,1]^{d_J}\times\mathcal{B}}\widetilde{\phi}_n(t_J,t_I)\widetilde{Z}(t_J,g(t_J)+t_I)\mathrm{d}\mu(t_J,t_I)\\
&= \sum_{n=1}^{+\infty}\underbrace{r_I(\mathcal{B},\mathcal{B})r_I(\mathcal{B},\mathcal{B})^{-1}}_{I_K}\widetilde{\phi}_n(x_J,\mathcal{B})\int_{[0,1]^{d_J}\times\mathcal{B}}\widetilde{\phi}_n(t_J,t_I)\widetilde{Z}(t_J,g(t_J)+t_I)\mathrm{d}\mu(t_J,t_I)\\
&= \sum_{n=1}^{+\infty}\widetilde{\phi}_n(x_J,\mathcal{B})\int_{[0,1]^{d_J}\times\mathcal{B}}\widetilde{\phi}_n(t_J,t_I)\widetilde{Z}(t_J,g(t_J)+t_I)\mathrm{d}\mu(t_J,t_I).
\end{aligned}$$

I_K is the identity matrix of size $K\times K$. The expectation becomes (for (x_J,x_I) in $[0,1]^{d_I}$):

$$\mathbb{E}\left[\widetilde{Z}(x_J,x_I) \mid \widetilde{Z}(t_J,g(t_J)+t_I)\ \forall (t_J,t_I)\in[0,1]^{d_J}\times\mathcal{B}\right] = r_I(x_I,\mathcal{B})r_I(\mathcal{B},\mathcal{B})^{-1}\widetilde{Z}(x_J,g(x_J)+\mathcal{B}).$$

- *Finally, the P process is equal to*

$$Z^P(x_J,x_I) = \widetilde{Z}(x_J,x_I)-r_I(x_I,\mathcal{B})r_I(\mathcal{B},\mathcal{B})^{-1}\widetilde{Z}(x_J,g(x_J)+\mathcal{B})\quad\forall (x_J,x_I)\in[0,1]^{d_J}\times[0,1]^{d_I}.$$

■

9.4 Fomulae and algorithms

9.4.1 Formulae of the EM procedure

This subsection gives the formulae of the $\mathcal{Q}_n$ involved in the optimization problems (4.4) used to estimate the parameters η_n in the EM algorithm. $\mathbf{1}_{\widetilde{\mathbb{X}}_1}$ is the unit column vector of size $nrows\left(\widetilde{\mathbb{X}}_1\right)$. Denoting by $\mathbb{X}^1$ and $\mathbb{X}^2$ two DoE's, $0_{\mathbb{X}^1,\mathbb{X}^2}$ is the null matrix of size $nrows(\mathbb{X}^1)\times nrows(\mathbb{X}^2)$.

Using the expectation of a quadratic form formula to Gaussian vectors:

$$\left\{\begin{array}{lcl} \mathcal{Q}_1(\eta_1,\eta^*) & = & n_{\widetilde{\mathbb{X}}_1}\log\sigma_1^2+\log\left|\rho_{\theta_1}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right)\right| \\ & & +\frac{Tr\left(\rho_{\theta_1}(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1)^{-1}C_1(\eta^*)\right)}{\sigma_1^2} \\ & & +\frac{(E_1(\eta^*)-m\mathbf{1}_{\widetilde{\mathbb{X}}_1})'\rho_{\theta_1}(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1)^{-1}(E_1(\eta^*)-m\mathbf{1}_{\widetilde{\mathbb{X}}_1})}{\sigma_1^2}, \\ \forall n\in[\![2,N]\!], & & \\ \mathcal{Q}_n(\eta_n,\eta^*) & = & n_{\widetilde{\mathbb{X}}_n}\log\sigma_n^2+\log\left|\rho_{\theta_n}\left(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n\right)\right| \\ & & +\frac{Tr\left(\rho_{\theta_n}(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n)^{-1}C_n(\eta^*)\right)}{\sigma_n^2} \\ & & +\frac{E_n(\eta^*)'\rho_{\theta_n}(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n)^{-1}E_n(\eta^*)}{\sigma_n^2}, \end{array}\right.$$

with

$$\left\{\begin{array}{lcll} E_1(\eta^*) & = & m^*\mathbf{1}_{\widetilde{\mathbb{X}}_1} & \\ & & +(\sigma_1^*)^2\rho_{\theta_1^*}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right)Cov_{\eta^*}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}\left(\boldsymbol{y}-\mathbb{E}_{\eta^*}\left[\boldsymbol{Y}\right]\right), & \\ C_1(\eta^*) & = & (\sigma_1^*)^2\rho_{\theta_1^*}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right)-(\sigma_1^*)^2\rho_{\theta_1^*}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right)Cov_{\eta^*}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}(\sigma_1^*)^2\rho_{\theta_1^*}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right), & \\ & & & \\ E_n(\eta^*) & = & Cov_{\eta^*}\left(Z_n\left(\widetilde{\mathbb{X}}_n\right),\boldsymbol{Y}\right)Cov_{\eta^*}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}\left(\boldsymbol{y}-\mathbb{E}_{\eta^*}\left[\boldsymbol{Y}\right]\right), & \forall n\in[\![1,N]\!], \\ C_n(\eta^*) & = & \left(\sigma_n^*\right)^2\rho_{\theta_n^*}\left(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n\right) & \\ & & -Cov_{\eta^*}\left(Z_n\left(\widetilde{\mathbb{X}}_n\right),\boldsymbol{Y}\right)Cov_{\eta^*}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}Cov_{\eta^*}\left(\boldsymbol{Y},Z_n\left(\widetilde{\mathbb{X}}_n\right)\right). & \end{array}\right. \tag{9.10}$$

Partial analytical solution The optima $m^{(i+1)}$ and $\sigma_n^{(i+1)}$ ($n\in[\![1,N]\!]$) have analytical forms obtained by solving the system formed when the corresponding partial derivatives of the $\mathcal{Q}_n$ vanish. Finally, at each new iteration $i+1$, the goal is to find $\theta_n^{(i+1)}$ ($n\in[\![2,N]\!]$) solution of the following problem

$$\left\{\begin{array}{ll} \min\limits_{\theta_n} & n_{\widetilde{\mathbb{X}}_n}\log\left(\left(\sigma_n^{(i+1)}(\theta_n)\right)^2\right)+\log\left(\left|\rho_{\theta_n}\left(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n\right)\right|\right), \\ & \\ \text{with } \left(\sigma_n^{(i+1)}(\theta_n)\right)^2 & = \frac{Tr\left(\rho_{\theta_n}(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n)^{-1}C_n(\eta^{(i)})\right)+E_n(\eta^{(i)})'\rho_{\theta_n}(\widetilde{\mathbb{X}}_n,\widetilde{\mathbb{X}}_n)^{-1}E_n(\eta^{(i)})}{n_{\widetilde{\mathbb{X}}_n}}, \end{array}\right.$$

and $\theta_1^{(i+1)}$ solution of the following problem

$$\begin{cases} \min\limits_{\theta_1} & n_{\widetilde{\mathbb{X}}_1} \log\left(\left(\sigma_1^{(i+1)}(\theta_1)\right)^2\right) + \log\left(\left|\rho_{\theta_1}\left(\widetilde{\mathbb{X}}_1, \widetilde{\mathbb{X}}_1\right)\right|\right), \\ \text{with } \left(\sigma_1^{(i+1)}(\theta_1)\right)^2 & = & \frac{Tr\left(\rho_{\theta_1}(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1)^{-1}C_1(\eta^{(i)})\right)+(E_1(\eta^{(i)})-m^{(i+1)}(\theta_1)\mathbf{1}_{\widetilde{\mathbb{X}}_1})'\rho_{\theta_1}(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1)^{-1}(E_1(\eta^{(i)})-m^{(i+1)}(\theta_1)\mathbf{1}_{\widetilde{\mathbb{X}}_1})}{n_{\widetilde{\mathbb{X}}_1}}, \\ \text{and } m^{(i+1)}(\theta_1) & = & \frac{\mathbf{1}'_{\widetilde{\mathbb{X}}_1}\rho_{\theta_1}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right)^{-1}E_1(\eta^{(i)})}{\mathbf{1}'_{\widetilde{\mathbb{X}}_1}\rho_{\theta_1}\left(\widetilde{\mathbb{X}}_1,\widetilde{\mathbb{X}}_1\right)^{-1}\mathbf{1}_{\widetilde{\mathbb{X}}_1}}, \end{cases}$$

where $E_n(\eta^{(i)})$ and $C_n(\eta^{(i)})$ ($n \in [\![1, N]\!]$) are given in equation (9.10).

9.4.2 Algorithms of chapter 5

Algorithm 1 optimizes a nested design of higher dimension. Algorithm 2 optimizes a non-nested design of same dimension. In both algorithms, m_{opt} takes the successive values of the sequence $\mathbb{X}^{(l)}$. For any sample m, $m[R, C]$ denotes the submatrix of m located at rows indexed by the vector R and columns indexed by the vector C. $m[R,]$ denotes the submatrix located at rows indexed by R and containing all columns. Similarly, $m[, C]$ denotes the submatrix located at columns indexed by C and containing all rows. $\mathcal{U}(V)$ denotes the uniform distribution among V components.

9.5 Papers

9.5.1 Paper: Sampling strategies for metamodel enrichment and automative fan optimization [Henner et al., 2019]

This paper was written in the context of an ANR project called PEPITO which focused on sampling strategies for metamodels applied on industrial issues, and more precisely on the fan system of Valeo. The paper in itself tackles the definition of the geometric parameters, the construction of the input space controlled by geometric constraints, and the enrichment strategy of the training samples for the metamodels.

SAMPLING STRATEGIES FOR METAMODEL ENRICHMENT AND AUTOMOTIVE FAN OPTIMIZATION

M. Henner[1] - B. Demory[1] - T. Gonon[1] – C. Helbert[2]

[1]VALEO Thermal Systems - France

[2] Ecole Centrale de Lyon - Institut Camille Jordan - UMR CNRS 5208 - Ecully, France

ABSTRACT

The strategy aimed at reducing the development time for a fan, which is oriented through the use of large meta-models which can be re-used and enriched along time. Efforts have been brought in the parameterization of the geometry and in the simulation process which provides pressure rise, torque and consequently efficiency.

Numerical Designs of Experiments (DoE) are then conducted in an 11 factor problem to fill the space and to build a first kriging model. This latter assumes that the output is a gaussian field for which parameters are computed by maximum likelihood estimation based on the numerical runs. Uncertainties associated with any predicted values can then be assessed using the variance, and two small validation plans are used to measure statistically the errors.

The variance given by the model is further used to map the areas in the domain which would need additional sampling. Then, two strategies are tested to select the most relevant sampling points, the first one being to reduce the range of the parameter variation, and the other one being to select them according to turbomachine design rules which would have dismiss some factor combinations. These two methods for sequential enrichment of the response are then compared and can even be combined with a trend given on the pressure rise. Answers from the kriging models are then assessed again in terms of statistical errors. These strategies will be described in the proposed paper through model comparison and optimization results.

KEYWORDS

Fan system – Optimization – Kriging - Sampling - Design of Experiment

NOMENCLATURE

DoE	Design of experiment
RMSE	Root Mean Square of the prediction Error of the metamodel on a test sample

LIST OF SYMBOLS

A'	transpose of A, with A a matrix
$x=(x_1,...,x_{11})$	vector of design variables (input variables), called point
Y(x)	gaussian process, function of the design variables, used by the kriging model for the output
m	mean of the gaussian process Y
Z(x)	gaussian process with zero mean part of the writing of Y

k(x,x')	covariance kernel of Z
σ^2	variance of Z
θ	range of Z
$r_\theta(x, x')$	correlation kernel of Z
$\mathcal{L}(m,\sigma^2,\theta)$	likelihood function of m, σ^2, and θ
$\mathbf{x}^{(i)}$	i^{th} training point, or i^{th} test point
X	matrix of all training points, also called design matrix, its lines are the x^((i))
y	vector of output values associated with training points, or with test points
R	correlation matrix of the training points
r(x)	correlation vector between x and the training points
α_i	coefficient
$h_i(x_i)$	usual function applied to a design variable
RMSE	Root Mean Square of the prediction Error of the metamodel on a test sample
N	number of test points
n	number of training points
$f(x)$	computer code value
$\hat{m}(x)$	kriging prediction, also called Kriging mean
$\hat{\sigma}^2(x)$	variance of kriging prediction, also called kriging variance

INTRODUCTION

Specifications in automotive applications can cover a wide range of operating points since design, architecture and engine power are very different from one vehicle model to the other. Each new development for a fan involves therefore a new optimization process to reach nominal pressure rise and torque. Typical pressure and efficiency curves versus flow rate are presented in figure 1 with indications on operating points.

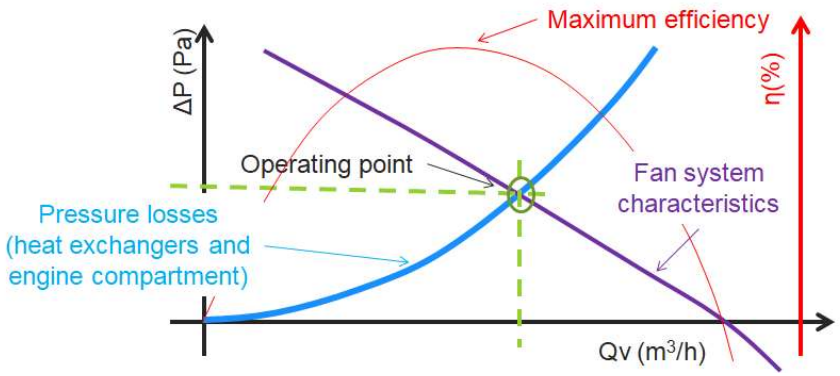


Figure 1: pressure and efficiency curves with operating points

Optimization is therefore perfectly adapted, and the current strategies aimed at reducing the development time for the fan, and are then oriented through the use of large meta-models which can be re-used and enriched along time. Investigations done on the current 11 parameters DoE are aimed to prepare a methodology for large dimensions, i.e. starting from 30 and going beyond, possibly up to 60. Accuracy must not be compromised neither by the space dimension (given by the number of parameters), nor by the widest range of possible variations. However, each simulation run has a non-neglectable cost which is not compatible with an intense sampling campaign. In the present case, 11 design variables have been selected and the full factorial plan would yield 2^{11} simulations, which is of course simply not possible.

Some strategies are then investigated in order to enrich initial Design of Experiment (DoE) which are classically based on a statistical approach uncorrelated from the studied physics. The question is to find a proper way to select the most relevant points which would improve the accuracy. However, one can object that the accuracy is only of interest for some limited set of parameters, when geometries are conforming to turbomachine laws. Therefore, in the context of a limited simulation budget, should the enrichment be based on a purely statistical approach, or is it worthy to use geometrical rules which exclude any sampling on *a priori* non interesting areas?

OBJECTIVES

The objectives of the current work are to investigate a sequential strategy for fan optimization, in the context of a limited simulation budget, which uses at first a meta-model from a kriging method, and secondly a genetic algorithm for optima research. Such methods have been intensively used and presented for low speed fans, for instance by Bamberger [1] or Verstraete [2].

Several strategies for the selection of potential additional runs are proposed and compared between them. Performances of these methods are assessed through different criteria measured on each surrogate model, which are given by global or local variances, by discrepancies with numerical results from either a global or a local validation plan, and by comparing the optimized final designs.

Geometrical parameterization of the fans and simulation processes are explained in the next paragraph, followed by some theoretical information on the kriging method which has been used. Sequential plans and methods for comparison are then presented before giving results and commenting on their respective efficiencies in the conclusion.

PARAMETERIZATION AND SIMULATIONS

Fan parameterization

A typical automotive fan is presented in figure 2. Several characteristics are frozen to standard values, and the selected parameters for optimization are the chord lengths and the stagger angles for five different radius. These ten geometrical factors are then completed by the flow rate which is imposed in the numerical experiment.

DoE must have independent parameters and ensure that all their combinations are feasible in the ranges of variation, meaning that no effect between them would prevent any geometry to be tested. However, it can lead to the creation of designs which can be said at least unconventional, or at worst opposite to the rule of turbomachine design. One can appreciate in figure 3, as an example a surprising resulting geometry on which the DoE can ask an evaluation of performances. In such cases, the heckled surfaces lead to chaotic flow patterns, yielding poor fan efficiencies and difficult numerical convergences (and consequently “signal” noise during the creation of meta-models). At a first thought, they are not representing the most interesting sampling points.

Numerical modeling

As presented in [3], performance predictions for the DoE are obtained, thanks to RANS simulations with the commercial code CCM+, using polyhedra meshes and an accurate description of the test rig (figure 4). The simulation process takes one hour of meshing (a new mesh is produced for each geometry), and requires between 4 to 8 wall clock time with 196 CPU (depending on the convergence). Mesh parameters are kept constant from one geometry to the other in order to minimize discrepancies. An accuracy assessment is provided in [3], and allow us to determine the mesh size above which variations remain below 1%. About 20 millions polyhedra are used, mainly

concentrated in the rotating domain (15 millions). The motion of the fan in the steady state run is modeled through multiple frames of references. Turbulence is modeled with the classical 2 equations model k-ω SST from Menter, and a two layers model is used to predict accurately the boundary layer on the walls. The layers of cell extruded from the surface mesh yield to an average wall y+ close to 1 on the fan blades. The flow rate is imposed at the inlet plenum of the test rig, whereas an atmospheric condition is imposed in the far field.

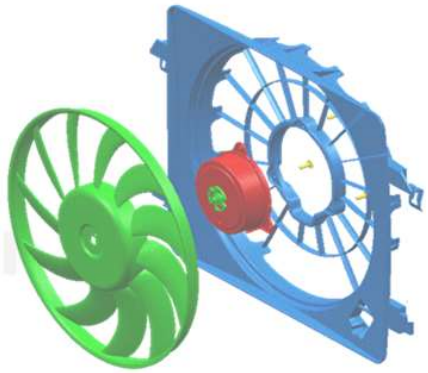


Figure 2: typical automotive fan

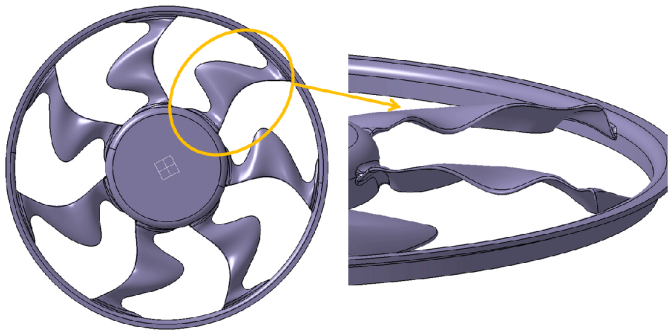


Figure 3: example of a surprising geometry

It allows us to monitor the performances with the pressure rise between an inlet and outlet (static to static) and the torque exerted by the fluid on the fan. Convergence is considered to be achieved when these values remain stable over a sufficient period of time, in addition to residues below 10^{-4}. Selected output responses are then the pressure rise (ΔP in Pa), the torque (T in N.m) and the global efficiency (%) deduced from the formula $\eta = (\Delta P * Q)/(T * \Omega)$, Q being the flow rate (m3/s) and Ω the rotational speed (rad/s).

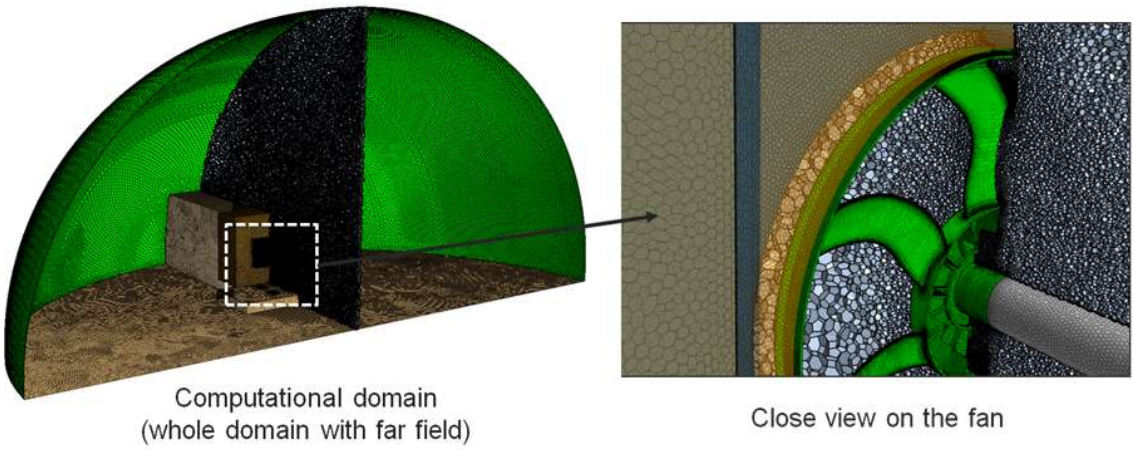


Figure 4: Simulation domain

DOE AND KRIGING METHOD

DOE's building [4]

All factors have been dimensionalized in the range [-1, 1], the set of data with 11 factors at zero being the center of the domain. An Orthogonal Latin Hypercube (OLH) plan has been selected for the initial sampling and the training of the meta-model. It is composed of 127 points that have been selected in the $[-1, 1]^{11}$ domain, according to the theory which imposes at first that each point must have a different value for each direction (a direction corresponding to an input variable), and secondly that the columns of the design matrix are orthogonal (one column contains the observed values of one design variable).

Two other Latin Hypercube Samples (LHS) designs (which were optimized for discrepancy) of 25 points each have been established for testing the meta-models trained from the 127 runs (and the points for enrichment if any). The first one is set in $[-1, 1]^{11}$, and the second one in $[-0.5, 0.5]^{11}$. For this later it means that all factors are limited in the range [-0.5, 0.5], i.e. only a central part of the domain is considered

Kriging meta-model building [5]

The kriging meta-model models the computer code, f, as a realization of a random field $\{Y(x), x \text{ in } [-1, 1]^{11}\}$. Each component of x is a particular input variable of the simulator. This random field is supposed to be such that:

$$Y(x) = m + Z(x) \quad (1)$$

where m is a scalar constant, called trend, and Z is a gaussian field with zero mean and a covariance function of the type :

$$k(x, x') = \sigma^2 r_\theta(x, x') \quad (2)$$

r_θ is called the correlation function. The one used in this work is a tensor product of the Matern $\frac{5}{2}$ kernel :

$$r_\theta(x, x') = \prod_{k=1}^{11}(1 + \frac{\sqrt{5}\left|x_k - x_k'\right|}{\theta_k} + \frac{5\left|x_k - x_k'\right|}{3\theta_k^2})\exp(-\frac{\sqrt{5}\left|x_k - x_k'\right|}{\theta_k}) \quad (3)$$

The Matern $\frac{5}{2}$ kernel is the most commonly used because it is twice differentiable and it is the most realistic with respect to the actual simulation outputs. Other properties of gaussian kernel or exponential kernel are detailed for instance in [6].

The three parameters m, σ^2, and θ are determined by maximizing the likelihood function. θ is a vector of 11 components aimed to take into account of the anisotropy of the model. If we consider the set of training observations y, the likelihood function is:

$$\mathcal{L}(m, \sigma^2, \theta) = \mathbb{P}(Y = y | m, \sigma^2, \theta) = \left(\frac{1}{2\pi\sigma^2}\right)^{\frac{n}{2}} \left(\frac{1}{det(R)}\right)^{\frac{1}{2}} \exp(-\frac{(y - m1_n)' R^{-1} (y - m1_n)}{2\sigma^2}) \quad (4)$$

where $R = \left(r_\theta\left(x^{(i)}, x^{(j)}\right)\right)$ is the correlation matrix on the design points $x^{(1)},\ldots,x^{(n)}$. The matrix of the observations (whose lines are the $x^{(i)}$) is noted X, and is called design matrix. The estimation of m is different from the least square estimation. The least square estimation of m would be given by the formula: $\frac{\boldsymbol{y^{(1)}+\cdots+y^{(n)}}}{\boldsymbol{n}} = \frac{\mathbf{1}_n'\boldsymbol{y}}{\mathbf{1}_n'\mathbf{1}_n}$. However, the likelihood estimation of m is given by the formula: $\frac{1_n' R^{-1} y}{1_n' R^{-1} 1_n}$. The least square estimation is a particular case of the maximum likelihood estimation when the correlation matrix R is equal to the identity matrix.

Given that, the prediction is:

$$\hat{y}(x) = E[Y(x)|Y(X) = y] = m + r'(x)R^{-1}(y - m1_n) \quad (5)$$

where $r(x) = (r_\theta(x, x^{(i)}))_i$ is the vector containing the correlation between Y(x) and $Y(x^{(1)}),\ldots,Y(x^{(n)})$.

A kriging variance is also associated to each point of the input domain. It measures the uncertainty of the prediction at this point. The kriging variance is given by:

$$\hat{\sigma}^2(x) = \sigma^2 - r'(x)R^{-1}r(x) \quad (6)$$

Finally, if the main effects of the design variables x_k are near known functions $h_k(x_k)$, this information can be added to the model by using a trend equal to a linear combination of the main effects rather than a constant trend. The prediction becomes:

$$\hat{y}(x) = m + \sum_{k=1}^{n} \alpha_k h_k(x_k) + r'(x)R^{-1}(y - F\,\alpha) \quad (7)$$

where $F = (1_n\ H),\ H = \left(h_k\left(x_k^{(l)}\right)\right)_{l\in\{1,\ldots,n\},k\in\{1,\ldots,11\}}$, and $\alpha = (m\ \alpha_1 \ldots \alpha_n)'$.

The vector α is estimated by maximum likelihood. It is computed by the formula : $(F'R^{-1}F)^{-1}F'R^{-1}y$. The least square estimation would be : $(F'F)^{-1}F'y$. Again, the least square estimation is a particular case of the maximum likelihood estimation when the correlation matrix R is equal to the identity matrix.

The goal is to improve the accuracy by adding global information on the "shape" of the meta-model.

SAMPLING METHODS

A previous approach studied by Ribaud [7] consists of adding points in the area of interest, extracting them from the Pareto front and selecting them according to a robustness criterion. This method is relevant for a known optimization problem (i.e. when the process can be guided in a limited area of the solution domain), but has no benefit for the global improvement of the meta-model. One proposal from Zhang [8] consists of, in using a small training sample (for example OLH) and then in adding points. His method proved to be very fast because he could use a co-kriging technique that drew additional information from the first and second derivatives. Of course, the great difficulty there is to have access to these values, and it implies either to derive the Navier-Stokes equations [9][10] or to assess the derivative with a differentiation method [11].

Enrichment in a restricted hypercube

The first trial consists of selecting one round of the 10 points according to the variance criterion introduced in the equation (6), the objective being to find the points with the maximum variance, which can help to improve the model globally by filling the most uncertain areas. The corners (factors equal to 1 or -1) are systematically proposed (figure 5), since they are in the extrapolation zone where the variance becomes large. Unfortunately, it would be hopeless to try any global improvement by selecting the 2^{11} corners. Alternatively, adding information in the center could be of interest since it would help to improve the model in the 11 directions. By limiting the selection in the $[-0.5, 0.5]^{11}$ domain, it is observed that the first point is very close to the center (red point in figure 6), and then the other points are in the corners of the limited domain.

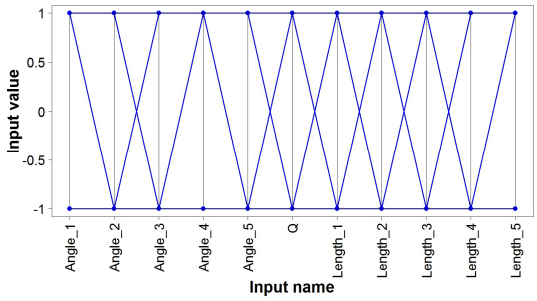


Figure 5. Added points with the maximal variance in $[-1, 1]^{11}$ for ΔP.

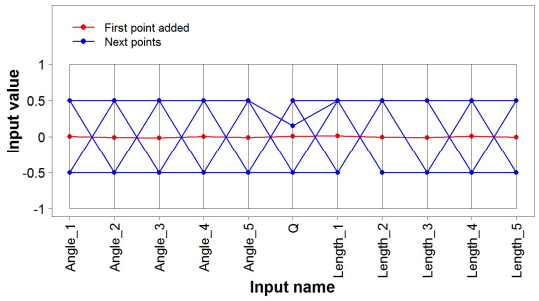


Figure 6. Added points with the maximal variance in $[-0.5, 0.5]^{11}$ for ΔP.

To compare the effect of adding point either in $[-1, 1]^{11}$ or in $[-0.5, 0.5]^{11}$, 10000 points from a Sobol sequence were generated and used to compare the mean of their kriging variance. However the mean of the variance criterion was assessed by removing the central point of this sequence because it is the first added point of the second method. This mean criterion is computed on the ΔP meta-model each time another point has been added at location of maximum kriging variance. The black curve (figure 7) shows the decrease of the mean variance when adding sequentially the points in $[-1, 1]^{11}$ whereas the red curve is for $[-0.5, 0.5]^{11}$. Initially, adding the central point is obviously worthy, and then the rate of improvement is rather equivalent.

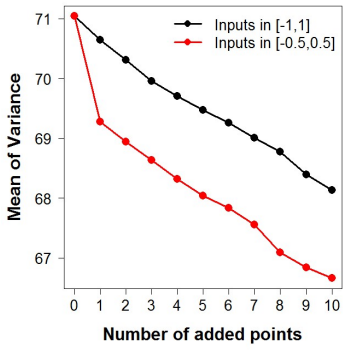


Figure 7. Comparison of the methods on the mean of variance criterion.

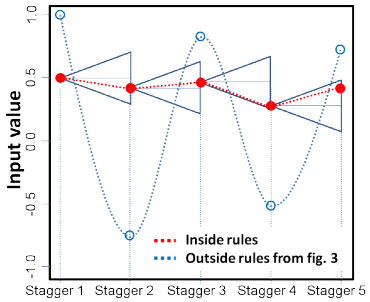


Figure 8: Examples satisfying the stagger angle constraints (red) or not satisfying them (blue, geometry from fig.3).

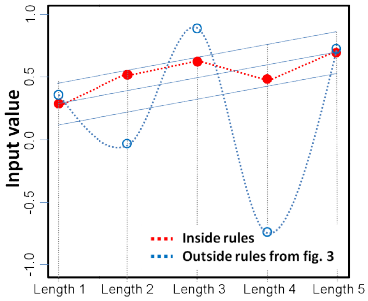


Figure 9: Examples satisfying the chord length constraints (red) or not satisfying them (blue, geometry from fig.3).

The first method to be evaluated in this article will be the one that adds points of maximum kriging variance in the restricted domain $[-0.5, 0.5]^{11}$.

Adding geometrical constraints

Points which are added must be relevant from a turbomachine point of view, in order to avoid strange geometries like in figure 3. When adding geometrical constraints, the selection process for the additional points simply discards the geometries which are not considered credible, and iterates until a validated point is found. The rules are simply based on the observation that irrelevant geometries are characterized by abrupt variations and large amplitudes of both chord length and stagger angles. It does not make sense to have angles of incidence which vary greatly from one radius to another, and that the load is not distributed smoothly along the blade span. Two rules are therefore applied and must be verified simultaneously, one concerning stagger angles and one concerning chord lengths.

The rules are defined to limit variations from one radius to another. The examples of figures 8 and 9 show possible evolutions, when variation for stagger angles are limited to +/-10%, and chord lengths do not deviate from the mean line between bottom and top by more than +/-10%. For the sake of comparison, the variations of the fan of figure 3 are presented on the graph. It is obvious that these former geometries don't follow these rules.

The second method to be evaluated in this article will be the one that adds points of maximum kriging variance among the points that satisfy geometrical constraints.

Adding a linear trend

As introduced by the formula (7), the trend can be defined by the following formula (given in eleven dimensions): $m + \sum_{k=1}^{11} \propto_k h_k(x_k)$. This method can be used by considering the pressure rise variation (ΔP) which is rather linear, and systematically decreasing versus flow rate (Qv). The linear trend will therefore have the following shape: $m + \alpha_Q Q$. In the article, the constant trend and the linear trend will be tested and compared for the ouput ΔP.

Comparing methods

The two validation plans (in $[-1, 1]^{11}$, and in $[-0.5, 0.5]^{11}$) are used to assess a criterion based on the Root Mean Square Error (RMSE), according to the formula:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y^{(i)} - \hat{m}(x^{(i)}))^2}{N}} \quad (8)$$

$x^{(i)}$ is the ith point of the test sample. $y^{(i)}$ is the true output value associated to $x^{(i)}$. $\hat{m}(x^{(i)})$ is the meta-model predicted output value associated to $x^{(i)}$. N is the number of points in the test sample.

Accuracy assessment in the Whole Domain

The objective in this paragraph is to compare the quality of two different strategies of enrichment which use only 10 additional points each. Of course these points are not the same according to the strategy. The two methods will be compared to the so-called "initial" model, for which the training sample is limited to the initial sample of 127 points. In the first method, ten points of maximum variance have been selected in $[-0.5, 0.5]^{11}$ and added to the initial training sample (forming a new training sample of 137 points). In the second model, ten points of maximum variance among points satisfying the geometrical rules have been added to the initial training

sample (forming another training sample of 137 points). In addition with these sampling strategies, linear and constant trends on the pressure curves have been compared.

In summary, there are 3 models with constant trend for the pressure rise, three others with linear trend for the same output, and 3 models for the torque (only constant trend on torque). Table 1 gives the comparison for initial model and the 2 sampling methods.

It can be observed that the enrichment methods have a low incidence on the RMSE criterion since the mean error remains at 12 Pa for the pressure rise, and at 0.03 N.m for the torque. Adding a trend is more promising since it improves the RMSE by 1 to 2 Pa. .

	RMSE on ΔP		RMSE on T
	With constant trend	With linear trend	
Initial	12 Pa	11 Pa	0.03 N.m
Enrichment in $[-0.5, 0.5]^{11}$	12 Pa	10 Pa	0.03 N.m
Enrichment with geometrical rules	12 Pa	11 Pa	0.03 N.m

Table 1: Error assessment of the meta-model in the whole domain

Accuracy assessment in the Middle of the Domain

Table 2 shows the results obtained with the same models than table 1, but evaluated in the domain $[-0.5, 0.5]^{11}$. Once again, the linear trend provides a rather good improvement of the RMSE of 1 to 3 Pa. Comparing the two sampling methods, it shows this time a noticeable difference: the geometrical rules even deteriorate the meta-model globally. However it does not state if it is relevant for an optimization which is supposed to go only in areas of interest for turbomachine design.

	RMSE on ΔP		RMSE on T
	With constant trend	With linear trend	
Initial	10 Pa	8 Pa	0.05 N.m
Enrichment in $[-0.5, 0.5]^{11}$	9 Pa	8 Pa	0.03 N.m
Enrichment with geometrical rules	19 Pa	16 Pa	0.06 N.m

Table 2: Error assessment of the meta-model in the middle of the domain

OPTIMIZATION

The meta-models are then compared on two optimization problems:

- Maximize ΔP, T=<1N.m, Q set at a given nominal flow rate
- Minimize T, ΔP>=250 Pa, Q set at a given nominal flow rate

Figures 10 and 11 show the set of data for the optimized solutions (maximizing ΔP), using kriging respectively without constant or linear trend. In both cases, the first two models are rather similar, i.e. with the initial DoE and the enrichment in the middle. At this point, it would indicate that the meta-model is already relevant and a small enrichment, despite being based on a statistical approach with variance, does not improve the result. In addition, the small improvement noticed on the RMSE with the trend does not really change the optimum proposed. For information, a view of an optimized fan is shown in figure 12.

The result is slightly different when using the meta-model which has been enriched with points fulfilling some criteria based on design rules. The set of data shows smaller variations in stagger angles and in chord length from a radius to another, indicating that the rules have modified the meta-model around their domains of validity. However, despite being less chaotic, the stagger angles still show a big variation from radius 1 to 2 and from radius 4 to 5. For the chord length, there is still a big change from radius 4 to 5. As a consequence, it must be noticed that the optimized solutions using the meta-model with rules are outside these rules.

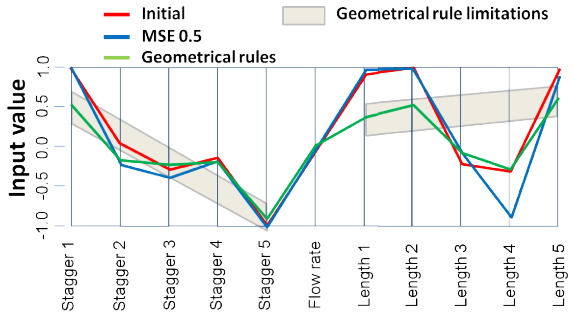


Figure 10: solutions when using ΔP meta-models with constant trend (first optim.)

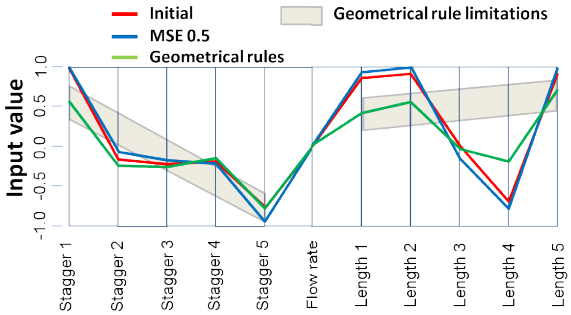


Figure 11: solutions when using ΔP meta-models with linear trend (first optim.)

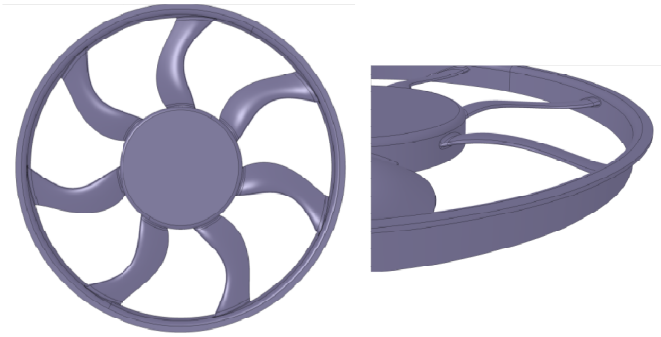


Figure 12: optimized fan

Errors given by the meta-models are presented in table 3, where the actual performances are those given by the simulation. The table shows the results obtained when using the linear trend in

the meta-model, however the results are quite similar without it as demonstrated by comparing figures 10 and 11. All proposed solutions are within a reasonable range of performance which is at the state of the art for such fans, i.e. close to 50% of static to static efficiency.

The enrichment in $[-0.5, 0.5]^{11}$, despite showing no RMSE improvement, has however brought a noticeable improvement in efficiency (+0.4% and +1.5%) versus the initial DoE. As no improvement is noticed in accuracy between these optimizations from the two meta-model, it would suggest that expecting accuracy is not the point, but rather having the good trend.

The last model which has introduced some rules for the choice of added points is finally the less performing one. Efficiencies are almost 3% below those obtained with the enrichment in the middle. Removing some points where the variance was important because of the geometrical criteria has probably lowered the ability of the kriging process to find good trends in the whole domain.

	Optim 1 (Maximize ΔP)		Optim 2 (Minimize T)	
	Values with linear trend	Errors with linear trend	Values with linear trend	Errors with linear trend
Initial	ΔP = 186 Pa T = 1.07 N.m E = **47.9 %**	$\Delta(\Delta P)$ = 11 Pa $\Delta(T)$ = 0.07 N.m $\Delta(E)$ = **6.4 %**	ΔP = 226 Pa T = 1.29 N.m E = 48.5 %	$\Delta(\Delta P)$ = 24 Pa $\Delta(T)$ = 0.06 N.m $\Delta(E)$ = **7.8 %**
Enrichment in $[-0.5, 0.5]^{11}$	ΔP = 190 Pa T = 1.08 N.m E = **48.3 %**	$\Delta(\Delta P)$ = 11 Pa $\Delta(T)$ = 0.08 N.m $\Delta(E)$ = 7.3 %	ΔP = 235 Pa T = 1.3 N.m E = **50 %**	$\Delta(\Delta P)$ = 15 Pa $\Delta(T)$ = 0.11 N.m $\Delta(E)$ = **7.8 %**
Enrichment with rules	ΔP = 173 Pa T = 1.06 N.m E = 45.1 %	$\Delta(\Delta P)$ = 27 Pa $\Delta(T)$ = 0.06 N.m $\Delta(E)$ = 10.0 %	ΔP = 218 Pa T = 1.28 N.m E = 47.1 %	$\Delta(\Delta P)$ = 32 Pa $\Delta(T)$ = 0.04 N.m $\Delta(E)$ = 8.4 %

Table 3: Actual performances for the optima, and errors of meta-model predictions

CONCLUSIONS

Investigations have been conducted in order to define the most relevant strategy for meta-modeling with sequential enrichment. It is assumed that the number of added points must be limited, and therefore a smart solution for optimizing the proposed runs must be found.

Two enrichment methods have been proposed, one being based on the variance in the center of the domain, and one adding some geometrical rules aimed to discard non conventional designs. In addition, a linear trend on the pressure rise has been included in the models and compared to the usual constant trend.

Accuracy assessment obtained with a RMSE criterion on validation designs has shown that limiting the sampling in the center of the domain seems to be an interesting strategy for scarce additional points, and that adding the trend improves the models. This would be all the more to be confirmed since tests with optimization have shown a small gain on maximum efficiency.

Conversely using geometrical rules for the enrichment does not help and even deteriorates the quality of both the meta-model and the optimization. It is still not explained, but presuming solutions is ultimately more counterproductive than letting the model learn from all points, including from those that are not good geometries.

ACKNOWLEDGMENTS

This work is being supported by the French Research Agency ANR [12], for the project PEPITO ("Plan d'Experience Pour l'Industrie du Transport et l'Optimisation")

REFERENCES

[1] K. Bamberger, T. Carolus, "*Optimization of axial fans with highly swept blades with respect to losses and noise reduction*", FAN 2012 Senlis (France) 18-20 April 2012

[2] T. Verstraete, P. Roytta, M. Henner, et al. "*Design and off-design optimization of a fan for automotive applications*". In: Proceedings of the 9th European conference on turbomachinery fluid dynamics and thermodynamics (ETC'9), Istanbul, Turkey, 2011, pp. 1485–1496.

[3] Henner M., Demory B., Franquelin F. and al., (2014). *"Test rig effect on performance measurement for low loaded high diameter fan for automotive application"*. ASME Turbo Expo, Frankfort, Germany, 2014.

[4] Qian P. Z., (2009). "*Nested latin hypercube designs*". Biometrika.

[5] Rasmussen C. E. and Williams C. K. I., (2006). "*Gaussian Processes for Machine Learning*". MIT Press.

[6] Loic Le Gratiet. "*Multi-fidelity Gaussian process regression for computer experiments"*. Autres [stat.ML]. Université Paris-Diderot - Paris VII, 2013. Français. 〈tel-00866770v2〉.

[7] Ribaud M., Blanchet C., Gillot F., and Helbert C., (2018). "*Robustness kriging-based optimization*". https://hal.archives-ouvertes.fr, <hal-01829889>

[8] Z. Zhang, B. Demory, M. Henner, B. demory, F. Franquelin, "*Space infill study of Kriging meta-model for multi-objective optimization of an engine cooling fan*", ASME Turbo Expo 2014, Dusseldorf, GT2014 June 16-20.

[9] L. Soulat, P. Ferrand, S. Moreau, S. Aubert, M. Buisson, "*Efficient optimisation procedure for design problems in fluid mechanics*", Computers & Fluids, Volume 82, 15 August 2013, Pages 73-86

[10] M. Buisson, P. Ferrand, L. Soulat, S. Aubert, S. Moreau, C. Rambeau, M. Henner, "*Optimal design of an automotive fan using the Turb-Opty meta-model*", Computers & Fluids, Volume 80, 10 July 2013, pp 207-213

[11] Z. Zhang, M. Buisson, P. Ferrand, M. Henner and F. Gillot, "*Meta-model based optimization of a large diameter semi-radial conical hub engine cooling fan*", Mechanics & Industry, Volume 16, Number 1, 2015.

[12]http://www.agence-nationale-recherche.fr/projetanr/?tx_lwmsuivibilan_pi2[CODE]=ANR-14-CE23-0011

Algorithm 1 Compute $\mathbb{X}_n$

Require: $\mathbb{X}^{(0)}$
  $m \leftarrow \mathbb{X}^{(0)}$
  $i \leftarrow 0$
  $T \leftarrow 10$
  $crit_m \leftarrow \Phi_n(m[[\![1, n_{\mathbb{X}_n}]\!],])$
  $m_{opt} \leftarrow m$
  $crit_{opt} \leftarrow crit_m$
  while $T > 0$ & $i < 200000$ **do**
    for $k = 1 : 10$ **do**
      $G \leftarrow m$
      $c \sim \mathcal{U}(\{1\} \cup I_n)$
      if $c = 1$ **then**
        $r_1 \sim \mathcal{U}\left([\![1, n_{\mathbb{X}_{n-1}}]\!]\right)$
        $r_2 \sim \mathcal{U}\left([\![1, n_{\mathbb{X}_{n-1}}]\!] \backslash r_1\right)$
        $G[r_1, I_1 \cup \cdots \cup I_{n-1}] \leftrightarrow G[r_2, I_1 \cup \cdots \cup I_{n-1}]$
      else
        $r_1 \sim \mathcal{U}([\![1, n_{\mathbb{X}_n}]\!])$
        $r_2 \sim \mathcal{U}([\![1, n_{\mathbb{X}_n}]\!] \backslash r_1)$
        $G[r_1, c] \leftrightarrow G[r_2, c]$
      end if
      $crit_G \leftarrow \Phi_n(G[[\![1, n_{\mathbb{X}_n}]\!],])$
      $p \leftarrow \min\left(\exp\left(\frac{crit_m - crit_G}{T}\right), 1\right)$
      if $p = 1$ **then**
        $m \leftarrow G$
        $crit_m \leftarrow crit_G$
        if $crit_m < crit_{opt}$ **then**
          $m_{opt} \leftarrow m$
          $crit_{opt} \leftarrow crit_m$
        end if
      else
        $b \sim \mathcal{B}(p)$
        if $b = 1$ **then**
          $m \leftarrow G$
          $crit_m \leftarrow crit_G$
        end if
      end if
    end for
    $i \leftarrow i + 1$
    $T \leftarrow \frac{T}{\log(i+2)}$
  end while
  $\mathbb{X}_n \leftarrow m_{opt}[[\![1, n_{\mathbb{X}_n}]\!],]$
  return $\mathbb{X}_n$

Algorithm 2 Compute $\mathbb{X}_{n,I_1}$

Require: $\mathbb{X}^{(0)} \subset [0,1]^{d_1}$ LHS of size $n_{\mathbb{X}_n}$
  $m \leftarrow \mathbb{X}^{(0)}$
  $i \leftarrow 0$
  $T \leftarrow 10$
  $crit_m \leftarrow \Phi_{n,1}(m)$
  $m_{opt} \leftarrow m$
  $crit_{opt} \leftarrow crit_m$
  while $T > 0$ & $i < 200000$ **do**
    for $k = 1 : 10$ **do**
      $G \leftarrow m$
      $c \sim \mathcal{U}(I_1)$
      $r_1 \sim \mathcal{U}(\llbracket 1, n_{\mathbb{X}_n} \rrbracket)$
      $r_2 \sim \mathcal{U}(\llbracket 1, n_{\mathbb{X}_n} \rrbracket \backslash r_1)$
      $G[r_1, c] \leftrightarrow G[r_2, c]$
      $crit_G \leftarrow \Phi_n(G)$
      $p \leftarrow \min\left(\exp\left(\frac{crit_m - crit_G}{T}\right), 1\right)$
      if $p = 1$ **then**
        $m \leftarrow G$
        $crit_m \leftarrow crit_G$
        if $crit_m < crit_{opt}$ **then**
          $m_{opt} \leftarrow m$
          $crit_{opt} \leftarrow crit_m$
        end if
      else
        $b \sim \mathcal{B}(p)$
        if $b = 1$ **then**
          $m \leftarrow G$
          $crit_m \leftarrow crit_G$
        end if
      end if
    end for
    $i \leftarrow i + 1$
    $T \leftarrow \frac{T}{\log(k+2)}$
  end while
  $\mathbb{X}_n^{(\mathcal{D}_0)} \leftarrow m$
  return $\mathbb{X}_n^{(\mathcal{D}_0)}$

9.5.2 Paper : Gaussian process regression on nested subspaces [Gonon et al., 2021]

This submitted paper deals with the main work of this thesis, which is to build a metamodel based on Gaussian process regression, which takes into account different DoE's, defined on nested subspaces of the input space, of increasing dimension.

Gaussian process regression on nested spaces

Thierry Gonon[1], Céline Helbert[1], Christophette Blanchet-Scalliet[1], and Bruno Demory[2]

[1]Univ Lyon, Ecole Centrale de Lyon, CNRS UMR 5208, Institut Camille Jordan, 36 avenue Guy de Collongue, F-69134 Ecully Cedex, France
[2]VALEO Thermal Systems, 8 rue Louis Lormand, 78321 La Verrière, France

Abstract Metamodels are widely used in industry to predict the output of an expensive computer code. As industrial computer codes involve a large amount of input variables, creating directly one big metamodel depending on the whole set of inputs may be a very challenging problem. Industrialists choose instead to proceed sequentially. They build metamodels depending on nested sets of variables (the variables that are set aside are fixed to nominal values), i.e. the dimension of the input space is progressively increased. However, at each step, the previous piece of information is lost as a new Design of Experiment (DoE) is generated to learn the new metamodel. In this paper, an alternative approach will be introduced, based on all the DoEs rather than just the last one. This metamodel uses Gaussian process regression and is called seqGPR (sequential Gaussian process regression). At each step n, the output is supposed to be the realization of the sum of two independent Gaussian processes $Y_{n-1} + Z_n$. The first one Y_{n-1} models the output at step $n-1$. It is defined on the input space of step $n-1$ which is a subspace of the one of step n. The second Gaussian process Z_n is a correction term defined on the input space of step n. It represents the additional information provided by the newly released variables. Z_n has the particularity of being null on the subspace where Y_{n-1} is defined so that there is a coherence between the steps. Firstly, some candidate Gaussian processes for $(Z_n)_{n\geqslant 2}$ are suggested, which have the property of being null on an infinite continuous set of points. Then, an EM (Expectation-Maximization) algorithm is implemented to estimate the parameters of the processes. Finally, the metamodel seqGPR is compared to a classic kriging metamodel where the output is assumed to be the realization of one second order stationary Gaussian process. The comparison is made on two analytic examples, a first one with two steps, up to dimension 4, and a second one with three steps, up to dimension 15. The introduced methodology is also tested on an industrial example which goes from dimension 11 to dimension 15. In all these test cases, seqGPR performs better than, or at least as well as kriging.

Keywords: Metamodel, Kriging, Gaussian process regression, High dimension, Infinite conditioning, Multifidelity, Nested spaces, Variable-size design space problems.

1 Introduction

In industry, numerical codes are widely used to model physical phenomena involved in manufacturing (see section 5.3 for an example of industrial product). Computer experiments make the search for performance easier and cheaper than physical experiments. However, computer codes are confronted to some limitations. As they model complex physical phenomena, they are often computationally expensive. This problem is solved by the use of metamodels [Forrester et al., 2008], [Sacks et al., 1989]. A metamodel is a simpler statistical model, like radial basis neural network [Cheng and Titterington, 1994], kriging model [Santner et al., 2003], [Roustant et al., 2012], support vector regression [Clarke et al., 2005]. This simpler model is fit on a sample of well-chosen runs, also called design of experiments (DoE) [Pronzato and Müller, 2012], [Dupuy et al., 2015]. Metamodels provide fast approximations of the code outputs. Besides, they have a proactive role as they help to select relevant simulations to be run [Jones et al., 1998]. The computer code is then used more efficiently.

Building surrogate models can be difficult when numerical codes involve lots of variables: geometrical parameters, environmental ones etc. Complex engineering systems can be found in [Makowski et al., 2006], [Lefebvre et al., 2010], [Auder et al., 2012]. Dimension reduction is a common approach to deal with it. It consists in considering only the influential inputs. The influential inputs are often detected by sensitivity analysis methods ([Saltelli, 2000],[Sobol, 1993],[Iooss and Prieur, 2019]). However, this approach leads to a vicious circle. Indeed, on one side, the variables selected by sensitivity analysis are relevant only if the

metamodel is sufficiently accurate, and on the other side, the dimension reduction is used to increase the metamodel accuracy. An alternative method that is usually chosen in industry is to act step by step. The first studies carried out on the code have focused on a small amount of variables deemed as important based on expert knowledge. The other variables are set to nominal values. A metamodel is created and used only for these variables until the understanding of their influence is sufficient. Then, to obtain better performances, finer studies are proceeded with, in which new variables are added progressively. The classical approach is to build a metamodel on a new independent space-filling design in the wider input space. Yet, this approach may be criticized as the information provided by the previous DoE's and metamodels is disregarded.

In the present work, an original metamodel is introduced, which takes into account the information from all the steps. The chosen approach consists in creating dependent metamodels from one step to another. The metamodels are based on Gaussian process regression [Santner et al., 2003] [Williams and Rasmussen, 2006]. One way of taking into account the different designs could be to use a multifidelity model for which the levels are not defined on the same input space. That is what is done in [Hebbal et al., 2021] with deep Gaussian processes. Nevertheless, this model is really expensive to learn. Furthermore, the multifidelity context is not suitable since the runs obtained at the previous steps have the same accuracy level. Another way could be to define a virtual categorical input, equal to the number of the step, that influences the choice of the input variables to consider. [Pelamatti et al., 2021] defines a Gaussian process whose input space varies dynamically with this virtual categorical input. This modelization does not take into account the fact that the input sets are entwined from one step to another. This particularity can lead to a simpler modelization. The approach in this work, called seqGPR (sequential Gaussian process regression), is based on a recursive statistical model inspired by the autoregressive multifidelity model [Kennedy and O'Hagan, 2000] but with the same accuracy for all the runs and with an input space of increasing dimension. The output of the code at a given step is modeled as the realization of a Gaussian process being the sum of the Gaussian process that models the previous step and a correction term. The correction term stands for the information provided by the new variables.

As regards this probabilistic model, one first difficulty is to build an appropriate correction term to represent the gap between the same numerical model observed on two nested subspaces. Thus, the correction process must be null on the subspace corresponding to the previous step. As this subspace is composed of an infinite continuous set of points, the question is then how to build a Gaussian process null on an infinite continuous set of points with a numerically computable covariance kernel. One idea could be to enjoin the prediction to verify the nullity property. For example, relevant points from the concerned subspace can be added to the training sample. This technique is used in [Da Veiga and Marrel, 2012] in the context of monotonicity, boundedness, convexity constraints. One drawback is the greediness of that method. Besides, the nullity cannot be verified in the whole subspace. Instead of establishing the nullity in retrospect, it can be verified at first glance, as an intrinsic property of the correction term. The correction process could be defined in a finite dimensional way, with adequate basis functions, as it is done in [Maatouk and Bay, 2017], [Lopera, 2019], [Bachoc et al., 2020] to ensure some properties of monotonicity, boundedness, convexity. Yet, this approach is greedy as the number of basis functions increases with the dimension. In this paper, the Gaussian process introduced in [Gauthier and Bay, 2012] which is an extension of the conditional expectation to an infinite continuous set of points is retained. A tractable kernel is sought for this process which is then compared to another candidate verifying the nullity condition.

Once a tractable candidate for the correction term is proposed, another difficulty lies in the estimation of the hyperparameters. One way is to optimize the likelihood. This task can be numerically difficult, since the likelihood involves several sets of hyperparameters for the different processes. One way to reduce the dimension of the optimization space is to propose nested designs as it is done in [Le Gratiet and Garnier, 2014]. In this paper an alternative based on the expectation maximization algorithm [Friedman et al., 2001] is introduced. It allows the reduction of the dimension with limited constraints on the designs.

The formulation of the metamodel seqGPR (sequential Gaussian process regression) is detailed in section 2. Two candidates for the correction term are proposed in section 3. An EM (expectation-maximization) algorithm is suggested in section 4 to estimate the model parameters. Finally, the method seqGPR is tested on two analytic examples and one industrial case in section 5.

2 Metamodel

Underlying processes Let $f : [0,1]^d \to \mathbb{R}$ be an output of an expensive simulation code depending on d variables. A response surface of f has to be created taking into account that $N-1$ previous studies of increasing dimension have already been completed. Each of these steps is a focus on the relation between the output and a subset of free variables. The other variables are temporarily fixed but then progressively released in the further steps. This paper deals with the handling of the last step, denoted by N, where all inputs are free.
Let I_n denote the index set of the variables that are released at step $n \in [\![1, N]\!]$ and $d_n = card(I_n)$. For every index set I, let x_I denote the associated subvector of x. The following framework, based on Gaussian process regression, is introduced to model the successive studies:

- At step 1, a first series of computer code evaluations is run. The set x_{I_1} of the first d_1 components of x, is free in $[0,1]^{d_1}$. Different values are explored. They are stored in DoE $\mathbb{X}_1 \subset [0,1]^{d_1}$. The other components are fixed to preset values $\dot{x}_{I_2 \cup \cdots \cup I_N}$ (entirely determined by x_{I_1}). Let f_1 denote the corresponding restriction of f on the subspace $[0,1]^{d_1}$. The function f_1 is assumed to be the realization of a stationary Gaussian process $Y_1 = m + Z_1$ of mean $m \in \mathbb{R}$ and with $Z_1 : \Omega \times [0,1]^{d_1} \to \mathbb{R}$, a centered Gaussian Process of covariance kernel $\sigma_1^2 \rho_1$. Let $\mathbf{y_1} = f_1(\mathbb{X}_1)$ represent the vector of observations of the output on DoE $\mathbb{X}_1$.

- At step 2, a second range of simulations is then launched on a subspace of a higher dimension $[0,1]^{d_1+d_2}$. The variables x_{I_2}, fixed at step 1, are released. Different values of $x_{I_1 \cup I_2}$, stored in DoE $\mathbb{X}_2$, are explored. The other variables $x_{I_3 \cup \cdots \cup I_N}$ are fixed to the new set of values $\dot{x}_{I_3 \cup \cdots \cup I_N} \in [0,1]^{d_3 + \cdots + d_N}$. Let f_2 be the corresponding restriction of f on the subspace $[0,1]^{d_1+d_2}$.
  The function f_2 is assumed to be the realization of a Gaussian process $Y_2 : \Omega \times [0,1]^{d_1+d_2} \to \mathbb{R}$, which is the sum of the process at step 1 Y_1, and a correction term Z_2, which represents the additional information provided by the released variables. As f_1 and f_2 are restrictions of the same function f, Y_1 and Y_2 have to coincide on the subspace defined at step 1. This results in the following definition of Y_2:
  $$Y_2(x_{I_1}, x_{I_2}) = Y_1(x_{I_1}) + Z_2(x_{I_1}, x_{I_2}), \quad \forall (x_{I_1}, x_{I_2}) \in [0,1]^{d_1+d_2},$$
  with Z_2 a centered Gaussian process independent from Z_1, of covariance kernel $\sigma_2^2 \rho_2$ such that for all x_{I_1} in $[0,1]^{d_1}$, $Z_2(x_{I_1}, \dot{x}_{I_2}) = 0$. Let $\mathbf{y_2} = f_2(\mathbb{X}_2)$ denote the vector of observations for this step.

- In the same way as previous steps, at step $n \in [\![3, N]\!]$, a DoE $\mathbb{X}_n$ is created in the higher space $[0,1]^{d_1 + \cdots + d_n}$. The variables $x_{I_1 \cup \cdots \cup I_{n-1}}$ and x_{I_n} are free and the others $x_{I_{n+1} \cup \cdots \cup I_N}$ are set to $\dot{x}_{I_{n+1} \cup \cdots \cup I_N} \in [0,1]^{d_{n+1} + \cdots + d_N}$. The corresponding restriction of f on this subspace, f_n, is evaluated on $\mathbb{X}_n$. The values are stored in $\mathbf{y_n} = f_n(\mathbb{X}_n)$. The function f_n is modeled as the realization of a Gaussian process $Y_n : \Omega \times [0,1]^{d_1 + \cdots + d_n} \to \mathbb{R}$. Following the same arguments as in step 2, Y_n is defined as:
  $$Y_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}) = Y_{n-1}(x_{I_1 \cup \cdots \cup I_{n-1}}) + Z_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}), \quad \forall (x_{I_1 \cup \cdots I_{n-1}}, x_{I_n}) \in [0,1]^{d_1 + \cdots + d_n},$$
  with Z_n a centered Gaussian process independent from $(Z_1, \cdots, Z_{n-1})$ of covariance kernel $\sigma_n^2 \rho_n$ and such that $Z_n(x_{I_1 \cup \cdots \cup I_{n-1}}, \dot{x}_{I_n}) = 0, \forall x_{I_1 \cup \cdots \cup I_{n-1}} \in [0,1]^{d_1 + \cdots + d_{n-1}}$.

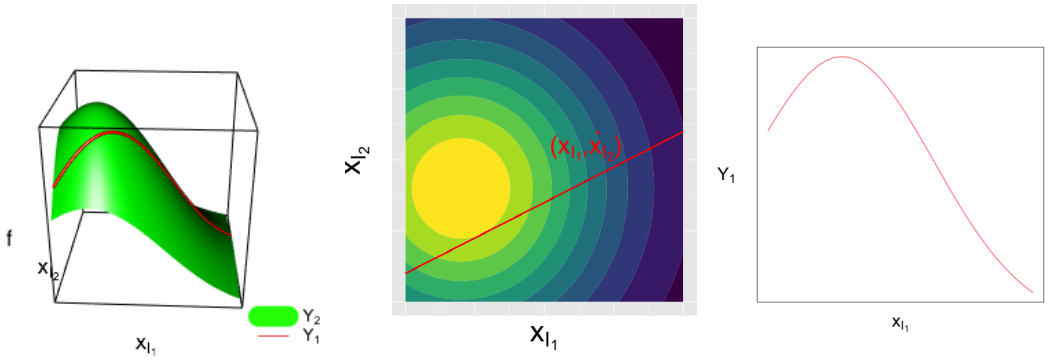
The coincidence of the $(Y_n)_{1 \leqslant n \leqslant N}$ on the subspaces is illustrated on figure 1 for $N = 2$.

Metamodel seqGPR Finally, the problem is modeled by the following statistical model:

$$\begin{cases} Y_1(x_{I_1}) = m + Z_1(x_{I_1}), & \forall x_{I_1} \in [0,1]^{d_1}, \\ Y_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}) = Y_{n-1}(x_{I_1 \cup \cdots \cup I_{n-1}}) + Z_n(x_{I_1 \cup \cdots \cup I_{n-1}}, x_{I_n}), & \begin{array}{l} \forall n \in [\![2, N]\!], \\ \forall x_{I_1 \cup \cdots \cup I_n} \in [0,1]^{d_1 + \cdots + d_n}, \end{array} \end{cases} \tag{1}$$

where:

- The processes $(Z_n)_{1 \leqslant n \leqslant N}$ are independent Gaussian processes of law:
  $$Z_n \sim \mathcal{GP}(0, \sigma_n^2 \rho_n(x_{I_1 \cup \cdots \cup I_n}, t_{I_1 \cup \cdots \cup I_n})) \quad \forall n \in [\![1, N]\!]. \tag{2}$$



(a) Visualization of Y_2 and Y_1 (b) Isocontours of Y_2 (c) Visualization of Y_1

Figure 1: Illustration of the model for $N = 2$. f is the realization of Y_2 (see panels on the left and center). On the cross section of equation $x_{I_2} = \grave{x}_{I_2}$, $Y_2 = Y_1$ (see panels on the left and right)

- The processes $(Z_n)_{2 \leqslant n \leqslant N}$ verify the following property:

$$Z_n(x_{I_1 \cup \cdots I_{n-1}}, x_{I_n}) \quad = \quad 0, \quad \forall n \in [\![2, N]\!], \forall x_{I_1 \cup \cdots I_{n-1}} \in [0,1]^{d_1 + \cdots + d_{n-1}}. \tag{3}$$

The same formulas of prediction as for a classic kriging metamodel apply. For every $x \in [0,1]^d$, the prediction mean $\hat{y}(x)$ and variance $\hat{v}(x)$ are defined by

$$\left\{ \begin{array}{rcl} \hat{y}(x) & = & \mathbb{E}\left[Y_N(x) \mid Y_1(\mathbb{X}_1) = \mathbf{y_1}, \cdots, Y_N(\mathbb{X}_N) = \mathbf{y_N}\right], \\ \hat{v}(x) & = & Var\left(Y_N(x) \mid Y_1(\mathbb{X}_1) = \mathbf{y_1}, \cdots, Y_N(\mathbb{X}_N) = \mathbf{y_N}\right). \end{array} \right.$$

Issues This model raises two issues. The first difficulty is to build $(Z_n)_{2 \leqslant n \leqslant N}$ such that it verifies the nullity property (cf (3)). This issue is tackled in section 3. Secondly, the estimation of the parameters of Y_1 and $(Z_n)_{2 \leqslant n \leqslant N}$ at step N, in the presence of training samples from the different steps, stands ahead as a thorny hurdle to overcome. It is detailed in section 4.

3 Candidates for the correction process

This section answers the first issue, which is to build a process that is null on an infinite continuous set of points. More precisely, the goal is to build a process $Z : [0,1]^{d_J + d_I} \times \Omega \to \mathbb{R}$ such that:

$$Z(x_J, g(x_J)) \quad = \quad 0 \quad \forall x_J \in [0,1]^{d_J}, \tag{4}$$

with $g : [0,1]^{d_J} \to [0,1]^{d_I}$ a deterministic function. In this section, two candidates are suggested for the process Z, both based on a latent Gaussian process $\widetilde{Z} \sim \mathcal{GP}\left(0, \sigma^2 r((x_J, x_I), (t_J, t_I))\right)$ of covariance kernel $\sigma^2 r$.

3.1 Red (reduced) process

A natural idea to obtain the nullity property (4) is to subtract from $\widetilde{Z}$ its value on the subspace of $[0,1]^{d_J + d_I}$ equal to $\{(x_J, g(x_J)) \mid x_J \in [0,1]^{d_J}\}$. The first candidate is:

$$Z^{Red}(x_J, x_I) = \widetilde{Z}(x_J, x_I) - \widetilde{Z}(x_J, g(x_J)).$$

It is called the Red process. It is a centered Gaussian process of covariance kernel $\sigma^2 \rho^{Red}$ with:

$$\begin{array}{rcl} \rho^{Red}((x_J, x_I), (t_J, t_I)) & = & r\left((x_J, x_I), (t_J, t_I)\right) - r\left((x_J, x_I), (t_J, g(t_J))\right) \\ & & -r\left((x_J, g(x_J)), (t_J, t_I)\right) + r\left((x_J, g(x_J)), (t_J, g(t_J))\right). \end{array}$$

3.2 P (preconditioned) process

Another way to obtain the nullity property (4) is to use the "conditional expectation" [Gauthier and Bay, 2012]. The second candidate is defined by

$$Z^P(x_J, x_I) \quad = \quad \widetilde{Z}(x_J, x_I) - \mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid \widetilde{Z}(s_J, g(s_J)), \forall s_J \in [0,1]^{d_J}\right], \ \forall (x_J, x_I) \in [0,1]^{d_J+d_I}. \tag{5}$$

It is called the P process. The term $\mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid \widetilde{Z}(s_J, g(s_J)), \forall s_J \in [0,1]^{d_J}\right]$ is the orthogonal projection of $\widetilde{Z}(x_J, x_I)$ in the sub Gaussian space engendered by the $\left(\widetilde{Z}(s_J, g(s_J))\right)_{s_J \in [0,1]^{d_J}}$. Its expression is given by [Gauthier and Bay, 2012]:

$$\mathbb{E}\left[\widetilde{Z}(x_J, x_I) \mid \widetilde{Z}(s_J, g(s_J)), \forall s_J \in [0,1]^{d_J}\right] \quad = \quad \sum_{n=1}^{+\infty} \phi_n(x_J, x_I) \int_{[0,1]^{d_J}} \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(s_J)) \mathrm{d}s_J, \tag{6}$$

with:

$$\phi_n(x_J, x_I) \quad = \quad \tfrac{1}{\lambda_n} \textstyle\int_{[0,1]^{d_J}} \sigma^2 r((x_J, x_I), (s_J, g(s_J))) \widetilde{\phi}_n(s_J) \mathrm{d}s_J,$$

and $(\lambda_n, \widetilde{\phi}_n)_{n \geqslant 1}$ are solutions of the eigen problem:

$$\int_{[0,1]^{d_J}} \sigma^2 r\left((x_J, g(x_J)), (s_J, g(s_J))\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J = \lambda_n \widetilde{\phi}_n(x_J), \ \forall x_J \in [0,1]^{d_J},$$

such that:

$$\int_{[0,1]^{d_J}} \widetilde{\phi}_n(s_J) \widetilde{\phi}_m(s_J) \mathrm{d}s_J = \delta_{nm}, \ \forall n, m \geqslant 1,$$

where δ_{nm} is the Kronecker symbol. Z^P is a centered Gaussian process of covariance kernel:

$$\sigma^2 \rho^P((x_J, x_I), (t_J, t_I)) \quad = \quad \sigma^2 r((x_J, x_I), (t_J, t_I)) - \sum_{n=1}^{+\infty} \lambda_n \phi_n(x_J, x_I) \phi_n(t_J, t_I), \quad \begin{array}{l} \forall (x_J, x_I) \in [0,1]^{d_J} \times [0,1]^{d_I}, \\ \forall (t_J, t_I) \in [0,1]^{d_J} \times [0,1]^{d_I}. \end{array}$$

This kernel cannot be used just as it is in practice, because the solutions of the eigen problem are not explicit in general and the sums are infinite. In this paper, two ways of computing this kernel are proposed. The first is based on the discretization of the spectral decomposition. The second is based on a wise choice of r (the correlation kernel of the latent process $\widetilde{Z}$) for which an explicit formula is known.

Numerical approximation In this paragraph, a first way of computing the kernel is given by an approximation based on the discretization of the spectral decomposition that reduces the functional eigen problem to a finite dimensional one. In that case, the terms from the spectral decomposition vanish and the formula of the resulting approximate kernel only depends on the kernel of the latent process.

Proposition 1 *Let $\mathbb{D} = \left(t_J^{(i)}, g(t_J^{(i)})\right)_{1 \leqslant i \leqslant L}$ be a uniform sample in the subspace $\{(t_J, g(t_J)), t_J \in [0,1]^{d_J}\}$. Then, discretizing the spectral decomposition of the P process Z by a Monte-Carlo method using $\mathbb{D}$ is equivalent to approximating the P process by the process $\widetilde{Z}$ conditioned to be null on the points of $\mathbb{D}$:*

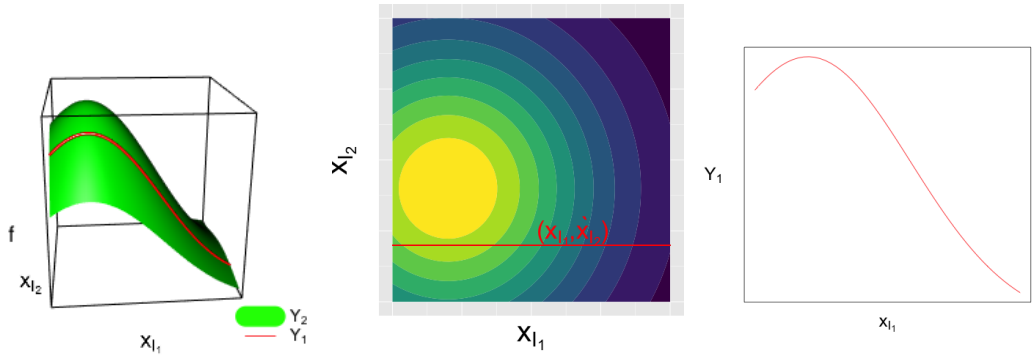
$$Z^{\mathbb{D}} = \left[\widetilde{Z} \mid \widetilde{Z}(\mathbb{D}) = 0\right].$$

$Z^{\mathbb{D}}$ is a centered Gaussian process of covariance kernel $\sigma^2 \rho^{\mathbb{D}}$:

$$\rho^{\mathbb{D}}(x, t) = r(x, t) - r(x, \mathbb{D}) r(\mathbb{D}, \mathbb{D})^{-1} r(\mathbb{D}, t) \quad \forall x, t \in [0,1]^{d_J+d_I}.$$

See appendix A for the proof of this proposition.

This process does not match the original expectations as the nullity is not fully respected on the continous set of points. The size of $\mathbb{D}$ must be high to correctly approach the full nullity. It is more and more difficult as the dimension of the input space increases, and it leads to very expensive computations (with inversion of huge matrices). Therefore, a second way of computing this kernel is proposed. It consists in finding forms of the kernel of $\widetilde{Z}$ that enable the kernel of Z^P to be tractable.



(a) Visualization of Y_2 and Y_1 (b) Isocontours of Y_2 (c) Visualization of Y_1

Figure 2: Illustration of the model for $N = 2$ (see figure 1), when $\dot{x}_{I_2}$ is constant.

Analytical expression in the case of a product kernel

Proposition 2 *Let r_J and r_I be two stationary correlation kernels. If r is of the form:*

$$r((x_J, x_I), (t_J, t_I)) = r_J(x_J, t_J) r_I(x_I - g(x_J), t_I - g(t_J)),$$

then the P process is a centered Gaussian process equal to:

$$Z^P(x_J, x_I) = \widetilde{Z}(x_J, x_I) - r_I(x_I - g(x_J), 0)\widetilde{Z}(x_J, g(x_J)),$$

whose covariance kernel is $\sigma^2 \rho^P$ with:

$$\rho^P((x_J, x_I), (t_J, t_I)) = r_J(x_J, t_J) \left[r_I(x_I - g(x_J), t_I - g(t_J)) - r_I(x_I - g(x_J), 0) r_I(0, t_I - g(t_J)) \right].$$

See appendix B for the proof of this proposition.

Corollary 1 *If g is constant equal to $c \in [0,1]^{d_I}$ and if r is a stationary kernel of the form*

$$r((x_J, x_I), (t_J, t_I)) = r_J(x_J, t_J) r_I(x_I, t_I),$$

then the P process is a centered Gaussian process equal to:

$$Z^P(x_J, x_I) = \widetilde{Z}(x_J, x_I) - r_I(x_I, c)\widetilde{Z}(x_J, c),$$

whose covariance kernel is $\sigma^2 \rho^P$ with:

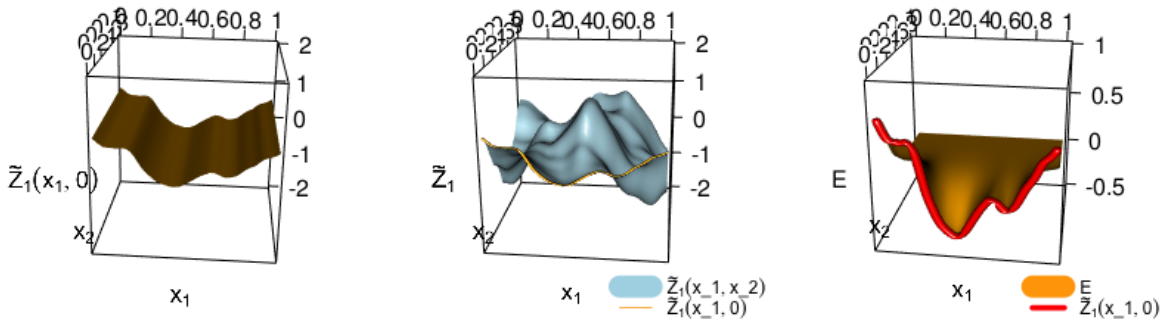
$$\rho^P((x_J, x_I), (t_J, t_I)) = r_J(x_J, t_J) \left[r_I(x_I, t_I) - r_I(x_I, c) r_I(c, t_I) \right].$$

The model presented in the corollary (where the released variables were previously fixed at a constant value) is illustrated in figure 2.

In the following, the P processes are assumed to be built as in proposition 2 or corollary 1.

3.3 Interpretation of the candidates

An illustration of the processes at stake in the construction of P and Red processes is shown in figure 3. They are used to build $Z(x_1, x_2)$ such that $Z(x_1, 0) = 0$. One main difference between Red and P process is that the initial latent process is disturbed locally in the case of the P process, whereas it is disturbed globally in the case of the Red process. Indeed, $\widetilde{Z}(x_1, 0)$ (see panel 3a), which is used to build the Red process, is constant in x_2 and consequently generates a disruption of $\widetilde{Z}(x_1, x_2)$ (see panel 3b) in the whole input space $[0,1]^2$. On the contrary, the conditional expectation $\mathbb{E}\left[\widetilde{Z}(x_1, x_2) \mid \widetilde{Z}(t_1, 0) \forall t_1 \in [0,1]\right]$ (see panel 3c), used to build the P process, depends on x_2, especially near the line $x_2 = 0$, and tends to 0 far from the line. The modification it implies on the latent process is therefore located almost only along the line.



(a) Value of latent process on the border $x_2 = 0$

(b) Initial latent process

(c) Conditional expectation of latent process

Figure 3: On middle panel, a realization of the latent process from which are built the two candidates for Z_1 that should be null on the line $x_2 = 0$. On left panel, the process which is removed from the latent process to create the Red process. On right panel, the process (denoted by E for expectation) which is removed from the latent process to create the P process.

4 Estimation of the parameters

This section answers the second issue which is to estimate the parameters of the metamodel seqGPR defined in section 2. An EM algorithm is presented to ease the maximization of the likelihood.

Notations At step N, the following training samples (also called observed data) are available: $(\mathbb{X}_1, \mathbf{y_1})$, ..., $(\mathbb{X}_N, \mathbf{y_N})$. The training data can be written as:

$$\begin{cases} Y_1(\mathbb{X}_1) & = & \mathbf{y_1}, \\ \vdots & & \\ Y_{N-1}(\mathbb{X}_{N-1}) & = & \mathbf{y_{N-1}}, \\ Y_N(\mathbb{X}_N) & = & \mathbf{y_N}, \end{cases}$$

or more simply:

$$\mathbf{Y} = \mathbf{y},$$

with:

$$\mathbf{Y} = \begin{pmatrix} Y_1(\mathbb{X}_1) \\ \vdots \\ Y_N(\mathbb{X}_N) \end{pmatrix} \quad \text{and} \quad \mathbf{y} = \begin{pmatrix} \mathbf{y_1} \\ \vdots \\ \mathbf{y_N} \end{pmatrix}.$$

The law of Y_1 (see equations (1) and (2)) depends on the parameters: m (scalar mean parameter), σ_1^2 (scalar variance parameter), and θ_1 (vector of covariance parameters). For all $n \in [\![2, N]\!]$, Z_n (see equation (2)) is either a Red or P process built on the latent process $\widetilde{Z}_n \sim \mathcal{GP}(0, \sigma_n^2 r_n)$ (see section 3) of parameters: σ_n^2 (scalar variance parameter), θ_n (vector of covariance parameters). In order to emphasize the dependence of ρ_n (the covariance kernel of Z_n, $n \in [\![1, N]\!]$) on the parameter θ_n, this kernel is denoted by ρ_{θ_n}. The purpose of this section is to estimate the parameters $\eta = (\underbrace{m, \sigma_1^2, \theta_1}_{\eta_1}, \underbrace{\sigma_2^2, \theta_2}_{\eta_2}, \cdots, \underbrace{\sigma_n^2, \theta_n}_{\eta_n}, \cdots, \underbrace{\sigma_N^2, \theta_N}_{\eta_N})$ based on the observed data. The maximum likelihood estimator is used. The following loss function [1] has to be minimized:

$$l(\eta; \mathbf{y}) = \log |Cov_\eta(\mathbf{Y}, \mathbf{Y})| + (\mathbf{y} - \mathbb{E}_\eta[\mathbf{Y}])' Cov_\eta(\mathbf{Y}, \mathbf{Y})^{-1} (\mathbf{y} - \mathbb{E}_\eta[\mathbf{Y}]),$$

with for any matrix M, M' denoting the transpose of M. This optimization problem is complex as η can reach big dimensions.

In what follows, the notation $\mathbb{X}_n \subset \mathbb{X}_{n-1}$ means that the submatrix of $\mathbb{X}_n$ composed of the columns corresponding to $x_{I_1 \cup \cdots \cup I_{n-1}}$ is included in $\mathbb{X}_{n-1}$. In this case, the designs $\mathbb{X}_n$ and $\mathbb{X}_{n-1}$ are said nested.

[1] equal to twice the negative log-likelihood up to a constant

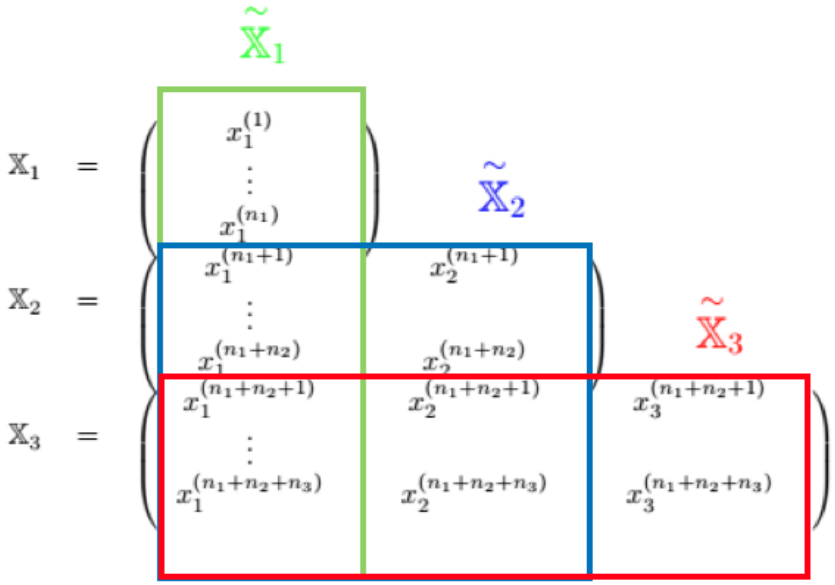


Figure 4: Definition of the $\widetilde{X}_n$ ($n \in \{1,2,3\}$) when $N = 3$ and $I_1 = \{1\}$, $I_2 = \{2\}$, $I_3 = \{3\}$

4.1 Nested designs

Proposition 3 *If all the designs are nested: $\mathbb{X}_N \subset \cdots \subset \mathbb{X}_1$, then the loss function can be decoupled:*

$$l(\eta; \mathbf{y}) = l_1(\eta_1; \mathbf{y_1}) + \sum_{n=2}^{N} l_n(\eta_n; \mathbf{z_n}),$$

with $l_1(\eta_1; \mathbf{y_1})$ (respectively $l_n(\eta_n; \mathbf{z_n})$, $n \in [\![1, N]\!]$) the loss function associated with the training data $Y_1(\mathbb{X}_1) = \mathbf{y_1}$ (respectively $Z_n(\mathbb{X}_n) = \mathbf{z_n}$).

Proof *$(\mathbf{z_n})_{1 \leqslant n \leqslant N}$ is observed because the designs are nested. The loss function can be decoupled because $(Y_1, Z_2, \cdots, Z_N)$ are independent.* ∎

If all the designs are nested, the parameters η_n can be estimated separately by optimizing the loss functions l_n.

4.2 Non nested designs

If the designs are not nested, the loss function is not decoupled. An EM (Expectation-Maximization, see [Friedman et al., 2001]) algorithm is then used to reduce the dimension of the optimization problem. To define this algorithm, the notion of complete data is introduced.

Definition 1 (Complete data) *Let $\widetilde{\mathbb{X}}_n$ ($\forall 1 \leqslant n \leqslant N$) denote the union of all training samples that concern Z_n. $\widetilde{\mathbb{X}}_n$ is a matrix of $d_1 + \cdots + d_n$ columns, composed of the concatenation of the parts of $\mathbb{X}_n, \cdots, \mathbb{X}_N$ corresponding to the input variables $x_{I_1 \cup \cdots \cup I_n}$. The complete data is the random vector: $(Y_1(\widetilde{\mathbb{X}}_1), Z_2(\widetilde{\mathbb{X}}_2), \cdots, Z_N(\widetilde{\mathbb{X}}_N))$.*

Figure 4 gives an illustration of the different $\widetilde{\mathbb{X}}_n$ for $N = 3$ and $I_1 = \{1\}$, $I_2 = \{2\}$, $I_3 = \{3\}$.

In the case of not nested designs $\widetilde{\mathbb{X}}_1, \cdots, \widetilde{\mathbb{X}}_N$ play the role of $\mathbb{X}_1, \cdots, \mathbb{X}_N$ in the case of nested designs. According to proposition 3, if the values of Y_1 on $\widetilde{\mathbb{X}}_1$, denoted by z_1, and the values of the Z_n on the $\widetilde{\mathbb{X}}_n$ ($n \in [\![2, N]\!]$), denoted by z_n, were observed, the loss function associated to all those training samples, denoted by l_c, could be decomposed as

$$l_c(\eta; z_1, \cdots, z_n) = \sum_{n=1}^{N} l_c^n(\eta_n; z_n),$$

where $l_c^1(\eta_1; \mathbf{z}_1)$ is the loss function associated to the training data $Y_1(\widetilde{\mathbb{X}}_1) = z_1$, and $l_c^n(\eta_n; \mathbf{z}_n)$ $(n \in [\![2, N]\!])$ is the loss function associated to the training data $Z_n(\widetilde{\mathbb{X}}_n) = z_n$.

As some of the complete data is not observed (missing data), an EM algorithm seems adequate to optimize the loss function.

Definition 2 (EM algorithm) *The EM algorithm is defined by*

- ***Expectation*** *Instead of the observed data loss function, the expectation of the complete data loss function conditioned by the observed data is considered. This quantity can be decomposed in N terms:*

$$\begin{aligned} \mathcal{Q}(\eta, \eta^*) &= \mathbb{E}_{\eta^*}\left[l_c(\eta; T) \mid \mathbf{Y} = \mathbf{y}\right], \\ &= \sum_{n=1}^{N} \mathcal{Q}_n(\eta_n; \eta^*), \end{aligned}$$

  with $\forall n \in [\![1, N]\!]$

$$\mathcal{Q}_n(\eta_n, \eta^*) = \mathbb{E}_{\eta^*}\left[l_c^n(\eta_n; Z_n(\widetilde{\mathbb{X}}_n)) \mid \mathbf{Y} = \mathbf{y}\right].$$

- ***Maximization*** *The EM algorithm consists in building the sequence $(\eta^{(i)})_{i \geqslant 0} = (\eta_1^{(i)}, \cdots, \eta_N^{(i)})_{i \geqslant 0}$ such that $\eta^{(0)}$ is user defined and $\forall i \geqslant 0$, $\eta^{(i+1)}$ is solution of the optimization problem*

$$\min_{\eta} \mathcal{Q}(\eta, \eta^{(i)}).$$

  For all $n \in [\![1, N]\!]$, $\eta_n^{(i+1)}$ is solution of the following optimization problem:

$$\min_{\eta_n} \mathcal{Q}_n(\eta_n, \eta^{(i)}). \tag{7}$$

See appendix C for the explicit formulas of the Q_n. It is known that the sequence $(\eta^{(i)})_{i \geqslant 0}$ verifies that $(l(\eta^{(i)}; \mathbf{y}))_{i \geqslant 0}$ is a decreasing sequence (see appendix D). Finally, η is estimated by a term of the sequence of sufficiently high rank, the choice of the rank being done by imposing a maximum number of iterations and a threshold on the variation of likelihood between iterations.

Remark To optimize $\mathcal{Q}_n$, the matrices $cov\left(Z_n(\widetilde{\mathbb{X}}_n), Z_n(\widetilde{\mathbb{X}}_n)\right) = \sigma_n^2 \rho_{\theta_n}(\widetilde{\mathbb{X}}_n, \widetilde{\mathbb{X}}_n)$ $(n \in [\![1, N]\!])$ need to be inverted. First, for all $n \in [\![1, N]\!]$, $\widetilde{\mathbb{X}}_n$ is assumed not having any redundant points such that the matrices have no identical rows. This condition is sufficient for the first matrix to be invertible as long as ρ_{θ_1} is a positive-definite kernel. As it is not the case of the ρ_{θ_n} $(n \geqslant 2)$, to ensure that the matrices are not singular, each $\widetilde{\mathbb{X}}_n$ $(n \in [\![2, N]\!])$ is also supposed to not contain any point on which Z_n is null so there are no rows full of zeros in the matrices.

5 Examples of application

In what follows, three metamodels are compared on three examples, two analytic functions and one output from an industrial code:

- seqGPR: the metamodel introduced in this paper. The kernel of Y_1 and the ones on which are based $(Z_n)_{n=2}^N$ are stationary tensor product matern $\frac{5}{2}$ covariance kernels.
- K_N: a classic kriging metamodel with a stationary tensor product matern $\frac{5}{2}$ covariance kernel and a constant mean which is built on the last training sample $(\mathbb{X}_N, \mathbf{y_N})$.
- K_tot: a classic kriging metamodel similar to K_N, but built on all training samples $(\mathbb{X}_1, \mathbf{y_1}), \cdots, (\mathbb{X}_N, \mathbf{y_N})$.

The philosophy of the method seqGPR is to explain a maximum of the data by Y_1, and to correct it thanks to the $(Z_n)_{n=2}^N$, so all the parameters for Y_1 are estimated individually and on the contrary some parameters of the Z_n are grouped. Different parameter sparsities for the Z_n are tried, to seek balance between fitting and robustness:

- Full: $\theta_n = (\theta_n^1, \cdots, \theta_n^{d_1+\cdots+d_n})$, one component θ_n^i for each input variable x_i.
- Robust: $\theta_n = (\underbrace{\theta_n^{I_1\cup\cdots\cup I_{n-1}}, \cdots, \theta_n^{I_1\cup\cdots\cup I_{n-1}}}_{I_1\cup\cdots\cup I_{n-1}}, \underbrace{\theta_n^1, \cdots, \theta_n^{d_n}}_{I_n})$, with $\theta_n^{I_1\cup\cdots\cup I_{n-1}}$ common to all components of $x_{I_1\cup\cdots\cup I_{n-1}}$, and one parameter θ_n^i for each component of x_{I_n}.

Four versions of seqGPR are compared. P_full and P_rob (resp. Red_full and Red_rob) use P (resp. Red) processes defined in subsection 3.2 (resp. 3.1), respectively with parameter sparsity of type Full and Robust for the $(Z_n)_{n=2}^N$. The metamodels are compared in terms of RMSE on a test sample. The training samples are not nested so the EM algorithm is used to estimate the parameters of the 4 versions of seqGPR. A simple likelihood estimation is used for the two kriging models.

5.1 Analytic example in dimension 4

The output f considered in this example is a function of 4 inputs $x = (x_1, x_2, x_3, x_4)$: $f(x_1, x_2, x_3, x_4) = f_1(x_1, x_2) + f_2(x_1, x_2, x_3, x_4)$, with:

$$\begin{cases} f_1(x_1, x_2) & = & \left[4 - 2.1(4x_1-2)^2 + \frac{(4x_1-2)^4}{3}\right](4x_1-2)^2 \\ & + & (4x_1-2)(2x_2-1) + \left[-4 + 4(2x_2-1)^2\right](2x_2-1)^2, \\ f_2(x_1, x_2, x_3, x_4) & = & 4\exp\left(-\|x - 0.3\|^2\right). \end{cases}$$

The study is composed of two steps:

- At step 1, computer code evalutations are run at DoE $\mathbb{X}_1$ in dimension 2, such that $f_1(\mathbb{X}_1) = \mathbf{y_1}$, with f_1 defined by $f_1(x_1, x_2) = f(x_1, x_2, \grave{x}_3, \grave{x}_4)$. Only the first two variables x_1 and x_2 are free and the other two variables are fixed: $\grave{x}_3 = \frac{x_1+x_2}{2}$, $\grave{x}_4 = 0.2x_1 + 0.7$.
- At step 2, new simulations are launched at points $\mathbb{X}_2$, a design in dimension 4. The last two variables (x_3, x_4) are now released.

The total number of variables is $d = 4$, the number of steps is $N = 2$, the index set of variables released at step 1 is $I_1 = \{1, 2\}$ and the index set of variables released at step 2 is $I_2 = \{3, 4\}$.

Figure 5 shows RMSE of the different methods computed on a Sobol sequence of size 10000 used as a test set. The RMSE is computed on 100 different training samples $(\mathbb{X}^1, \mathbf{y}^1)$ and $(\mathbb{X}^2, \mathbf{y}^2)$. The 100 RMSE's are represented by a boxplot for each metamodel. Different sizes of training sample are tested. The standard deviation of the output on the test set is represented by the black horizontal line.

All metamodel performances improve with the size of the training sample. Results show that including the previous information of $(\mathbb{X}^1, \mathbf{y}^1)$ is useful, as it greatly improves the performance of the kriging metamodel (K_tot is better than K_2). The robust seqGPR metamodels are better than the full, they are equivalent to or better than K_tot. It seems that the cases with a small or high number of training points are more discriminating than in the middle cases. With a small number of points (10pts-5pts), K_tot seems more destabilized than seqGPR. With a high number of points (20pts-40pts), seqGPR is more accurate than K_tot. Finally, seqGPR using Red are better than seqGPR using P. In the following examples, only the robust versions of seqGPR, which are more performing and whose parameter estimation is less complex, are compared to K_tot.

5.2 Analytic example in dimension 15

The following example should be less favorable to seqGPR. The objective function considered is in higher dimension and not decomposed as a sum of functions that respects the order in which the variables are released

$$f \quad : \left| \begin{array}{lll} [-3,3]^{15} & \to & \mathbb{R} \\ x & \mapsto & a_1' x + a_2' \sin x + a_3' \cos x + x' M x. \end{array} \right. \tag{8}$$

The function f was first used in [Oakley and O'Hagan, 2004]. The values of its coefficients a_1, a_2, a_3 and M can be found in www.sheffield.ac.uk/st1jeo. The inputs are rescaled in $[0, 1]$ and are rearranged by decreasing order of Sobol index.

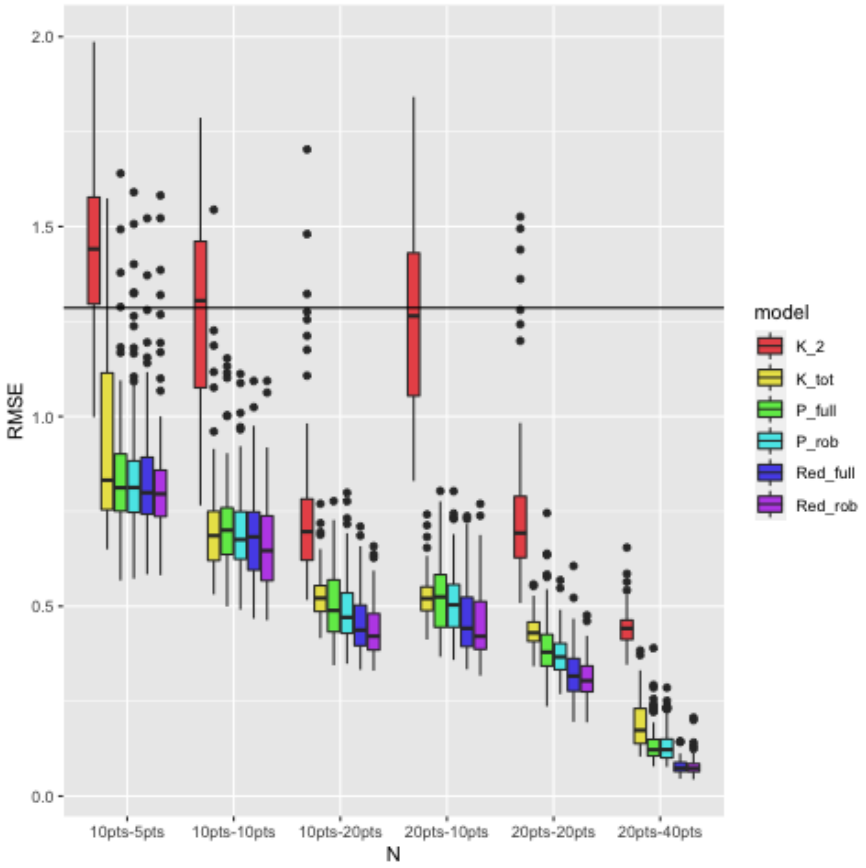


Figure 5: Boxplots of the metamodel RMSE's on 100 studies, for the analytic example in 4D. In abscissa are the sizes of the training samples.The following sizes are tested from left to right: $\mathbb{X}_1$ of size 10 and $\mathbb{X}_2$ of size 5 (10pts-5pts), $\mathbb{X}_1$ of size 10 and $\mathbb{X}_2$ of size 10 (10pts-10pts), $\mathbb{X}_1$ of size 10 and $\mathbb{X}_2$ of size 20 (10pts-20pts), $\mathbb{X}_1$ of size 20 and $\mathbb{X}_2$ of size 10 (20pts-10pts), $\mathbb{X}_1$ of size 20 and $\mathbb{X}_2$ of size 20 (20pts-20pts), $\mathbb{X}_1$ of size 20 and $\mathbb{X}_2$ of size 40 (20pts-40pts)

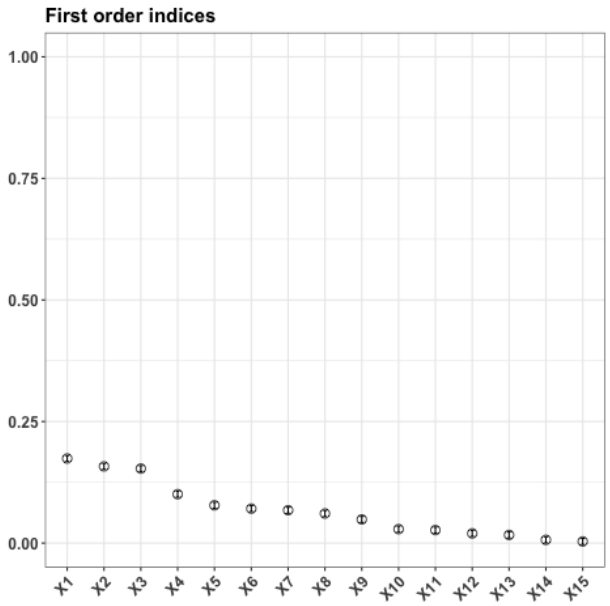
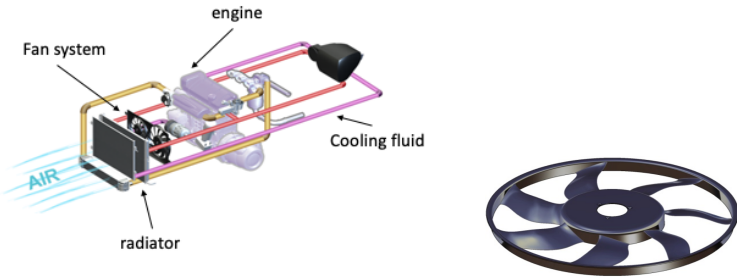


Figure 6: Sobol indices of the objective function (8).

	K_tot	P_rob	Red_rob
Median ($\cdot 10^{-4}$)	46.039	37.135	**36.903**
Interquartile range ($\cdot 10^{-4}$)	7.258	6.660	**6.570**

Table 1: Performance of the metamodels for the 100 studies made on the analytic 15D example. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$).



(a) Car engine cooling system (b) Fan system

Figure 7: On left panel, diagram of the car engine cooling system. On right panel, diagram of fan system. [Valeo,]

In practice, the first studies are done on the most influential inputs, chosen according to experts' physical knowledge. Here, as the function is analytical, the choice of the releasing order for the inputs is made by sensitivity analysis, which is directly performed on the function. Following the distribution of the Sobol indices (see figure 6), a study of $N = 3$ steps is made, adding respectively the groups of variables $I_1 = \{1, 2, 3\}$, $I_2 = \{4, 5, 6, 7, 8, 9\}$, and $I_3 = \{10, 11, 12, 13, 14, 15\}$. The variables that are fixed are set to 0.5. Three training samples, one for each step, are generated with 5 points per dimension considered at the step: $\mathbb{X}_1 \subset [0, 1]^3$ of size 15, $\mathbb{X}_2 \subset [0, 1]^9$ of size 45, and $\mathbb{X}_3 \subset [0, 1]^{15}$ of size 75. A Sobol sequence of size 10000 in dimension 15 is used as a test sample. The study is done 100 times, each time with new training samples.

The performances of the metamodels are shown in table 1. All versions of seqGPR are better than the classic kriging, with a slight improvement for the Red version in comparison with the P version. Red and P are almost equivalent as opposed to the previous example. This can be due to the fact that the P version disturbs less the initial stationary kernel because the variables which are fixed at a given step are set to a constant instead of varying in function of the free variables, as it is the case in the previous example.

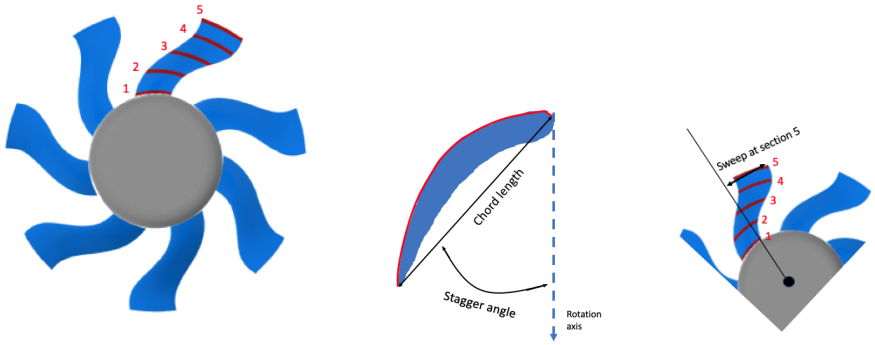
5.3 Industrial example: fan system in dimension 15

The industrial product which is studied in this section is the fan system which is part of a car engine cooling system from Valeo company (see figure 7). This cooling system is composed of a cold fluid circulating in part in the car engine to regulate its temperature and in part in a radiator where it is itself cooled down. When the car moves, the wind generated by the car speed and reaching the radiator is sufficient to evacuate the heat from the fluid. When the car is motionless, the fan is activated to replace the wind.

The output considered here is the Pressure difference (ΔP) between the upstream and the downstream of the air flow crossing the fan. It is function of 15 input variables. One input is the flowrate (Q). The 14 others are geometric. The fan blade geometries are supposed identical with each other. The geometry is monitored at 5 different sections (see figure 8a). At each section, the stagger angle and the chord length are controlled (see figure 8b). The chord is the line formed by the two borders of the blade at the considered section. The stagger angle is the angle between the chord and the rotation axis. The sweep is manipulated at each section. It is defined as the distance between the line formed by the rotation axis and the right border of the first section and the right border of the considered section (see figure 8c). At section 1, it is always equal to 0 and therefore not kept as an input.

To summarize, the variables are

- The flowrate: Q
- The stagger angles at the five sections: $Stag1$, $Stag2$, $Stag3$, $Stag4$, $Stag5$



(a) Sections

(b) Chord length and stagger angle at a given section

(c) Definition of the sweep at section 5

Figure 8: On the left panel, visualization of the five sections of the blade. On the middle panel, definition of the stagger angle and the chord length at a given section. On the right panel, definition of the sweep at section 5.

	K_tot	P_rob	Red_rob
Median ($\cdot 10^{-3}$)	44.585	**41.207**	42.088
Interquartile range ($\cdot 10^{-3}$)	5.892	4.657	**4.609**

Table 2: Performance of the metamodels for the 100 studies made on the industrial 15D example. The performances are shown in terms of median of RMSE's and interquartile range ($q_{75\%} - q_{25\%}$)

- The chord lengths at the five sections: $Chord1$, $Chord2$, $Chord3$, $Chord4$, $Chord5$
- The sweeps at sections 2 to 5: $Swe2$, $Swe3$, $Swe4$, $Swe5$

In the rest of this example, all variables are adimensionalized in $[0, 1]$ and the output ΔP is adimensionalized in $[-1, 1]$.

Industrial experts at Valeo have carried out a two-step ($N = 2$) study in the context of this work.

- At step 1, all the sweeps are fixed to constants: $Swe2 = 0.517645$, $Swe3 = 0.82$, $Swe4 = 1$, $Swe5 = 1$. An OLH of size 126 has been created on the 11 free inputs ($I_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$).
- At step 2, all the inputs are free ($I_2 = \{12, 13, 14, 15\}$). An OLH of size 299 has been created for the 15 inputs.

In order to obtain robust results, a cross-validation procedure is implemented. Each OLH is decomposed in 30 subsamples of size 50 optimized for the maximin criterion (they can share common points). The 30 complementary subsamples of size 249 from the OLH of step 2 are used as test samples.

Table 2 shows performances of the different metamodels on the test samples. The metamodel seqGPR is better than K_tot. Its version with P is this time slightly better than with Red, again probably because the fixed inputs are constant as in the previous example.

6 Conclusion

In the framework of Gaussian process regression, the problem of building metamodels in nested spaces of increasing dimension is studied. The variables are first fixed and then released progressively. The suggested method, called seqGPR, consists in creating a kriging metamodel with a non-stationary kernel on the reunion of all training samples. This kernel is the covariance kernel of a process defined recursively. At a given step, the process modeling the output is equal to the sum of the process modeling the output at the previous step and an independent correction term.

The correction term must be a Gaussian process which is null on a part of the input space. Two candidates are proposed: the Red (Reduced) process, and the P (Preconditioned) process. The P process, found in the

literature, is intractable. Two ways are tried to make it tractable. The first way is to discretize its spectral decomposition. But this technique leads to a Gaussian process conditioned to be null on some points of the space. This process is never exactly null everywhere in the part we want. To approach this nullity, a lot of points are needed in the conditioning, and that is very computationally expensive. In a second approach, a tractable exact expression is directly sought, in exchange for a modification of the latent process it is built upon.

Then, the issue of the parameter estimation is addressed. Instead of directly optimizing the likelihood, an EM algorithm is implemented, which is adapted to the additive structure composed of independent processes.

Finally, the metamodel seqGPR is compared to a classic kriging metamodel, based on a stationary kernel, on two analytic and one industrial examples. Different levels of parameter sparsity for the correction processes are tried. The level of sparsity that seems to always be better than classic kriging is the Robust one with either Red or P processes. The Red version seems to achieve higher performances than the P version most of the time, so the Red_rob version is recommended.

Software and Acknowledgments

Implementation have been done in R, where all the kernels and methods of estimation and prediction have been implemented using the package RcppArmadillo.

This work, carried out at Ecole Centrale de Lyon, was financially subsidized by Valeo. The authors wish to show their particular gratitude to Mr. Manuel Henner and Mr. Nicolas Yoan François for their support and fruitful discussion on the industrial point of view. The authors are grateful to the Members of GdRMascotnum for their useful advice, and in particular to Mr. Olivier Roustant and Mr. Xavier Bay for constructive discussions on the subject on the theoretical point of view. The authors would also like to acknowledge Mr. Philippe du Bouays and Ms. Laura Gay for their careful proofreading.

Appendix

A Proof of proposition 1

This section deals with the proof of proposition 1, which gives an approximation of the process Z^P (defined in equation (5)) by discretizing its spectral decomposition.

Proof • *Approximation of* $(\lambda_n, \widetilde{\phi}_n)$*:*

The following eigenvalue problem is considered:

$$\int_{[0,1]^{d_J}} \sigma^2 r((x_J, g(x_J)), (s_J, g(s_J)))\widetilde{\phi}(s_J)\mathrm{d}s_J \quad = \quad \lambda\widetilde{\phi}(x_J) \quad \forall x_J \in [0,1]^{d_J}.$$

It is discretized using a Monte-Carlo method. A sample $(s_J^{(i)})_{1\leqslant i\leqslant L}$ *is generated uniformly in* $[0,1]^{d_J}$ *to build* $\mathbb{D} = \left(s_J^{(i)}, g(s_J^{(i)})\right)_{1\leqslant i\leqslant L}$ *which is a discretization of the subspace* $\{(s_J, g(s_J)), s_J \in [0,1]^{d_J}\}$*. The Monte-Carlo approximation of the integral is:*

$$\frac{1}{L}\sum_{i=1}^{L} \sigma^2 r((x_J, g(x_J)), (s_J^{(i)}, g(s_J^{(i)})))\widetilde{\phi}(s_J^{(i)}).$$

Discretizing in x_J *with* $\mathbb{D}$*, the eigen problem becomes a finite dimensional one:*

$$\left(\frac{1}{L}\sigma^2 r(\mathbb{D}, \mathbb{D})\right)\widetilde{\Phi} = \gamma\widetilde{\Phi}. \tag{9}$$

The solutions of (9) *are noted* $(\gamma_n, V_n)_{1\leqslant n\leqslant L}$*. The* V_n *are taken such that they verify:*

$$V_n' \, V_m = \delta_{nm}.$$

The discretization of $\widetilde{\phi}_n$ *(noted* $\widetilde{\Phi}_n$*) must verify the discrete equivalent of*

$$\int_{[0,1]^{d_J}} \widetilde{\phi}_n(t_J)\widetilde{\phi}_m(t_J)\mathrm{d}t_J = \delta_{nm},$$

which is

$$\frac{1}{L}\ \widetilde{\Phi}_n'\ \widetilde{\Phi}_m = \delta_{nm}.$$

So the following relation between V_n *and* $\widetilde{\Phi}_n$ *is verified:* $V_n = \frac{1}{\sqrt{L}}\widetilde{\Phi}_n \Leftrightarrow \widetilde{\Phi}_n = \sqrt{L}V_n$.

- *Approximation of* ϕ_n*:*

  As

$$\phi_n(x_J, x_I) \quad = \quad \tfrac{1}{\lambda_n}\int_{[0,1]^{d_J}} \sigma^2 r\left((x_J, x_I), (s_J, g(s_J))\right)\widetilde{\phi}_n(s_J)\mathrm{d}s_J \quad \forall (x_J, x_I) \in [0,1]^{d_J}\times[0,1]^{d_I},$$

  using the same Monte-Carlo approximation and replacing $(\lambda_n, \widetilde{\phi}_n)$ *by* $(\gamma_n, \widetilde{\Phi}_n)$*, the following approximation of* ϕ_n *is obtained:*

$$\phi_n^{\mathbb{D}}(x) \quad = \quad \tfrac{1}{\gamma_n}\tfrac{1}{L}\sigma^2 r(x, \mathbb{D})\widetilde{\Phi}_n \quad \forall x \in [0,1]^{d_J+d_I}.$$

- *Decomposition of the matrix* $\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1}$

  $(\gamma_n, V_n)_{n\geqslant 1}$ *are the eigenvalues and (orthonormal) eigenvectors of* $\frac{1}{L}\sigma^2 r(\mathbb{D},\mathbb{D})$*. So* $(\frac{1}{\gamma_n}, V_n)_{n\geqslant 1}$ *are the eigenvalues and (orthonormal) eigenvectors of* $L\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1}$*, and:*

$$\begin{aligned} L\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1} &= \sum_{n=1}^{L}\frac{1}{\gamma_n}V_n V_n', \\ &= \sum_{n=1}^{L}\frac{1}{\gamma_n}\left(\frac{1}{\sqrt{L}}\widetilde{\Phi}_n\right)\left(\frac{1}{\sqrt{L}}\widetilde{\Phi}_n'\right), \\ &= \frac{1}{L}\sum_{n=1}^{L}\gamma_n\widetilde{\Phi}_n\widetilde{\Phi}_n'. \end{aligned}$$

  So

$$\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1} = \frac{1}{L^2}\sum_{n=1}^{L}\gamma_n\widetilde{\Phi}_n\widetilde{\Phi}_n'.$$

- *Approximation of the process* Z^P*:*

  The integral $\int_{[0,1]^{d_J}}\widetilde{\phi}_n(t_J)\widetilde{Z}(t_J, g(t_J))\mathrm{d}t_J$*, involved in the formula of* Z^P *(see* (5)*), is discretized the same way as before. It becomes:*

$$\frac{1}{L}\widetilde{\Phi}_n'\widetilde{Z}(\mathbb{D}).$$

  The approximation of the process is:

$$\begin{aligned} Z^{\mathbb{D}}(x) &= \widetilde{Z}(x) - \sum_{n=1}^{L}\phi_n^{\mathbb{D}}(x)\left(\frac{1}{L}\widetilde{\Phi}_n'\widetilde{Z}(\mathbb{D})\right), \\ &= \widetilde{Z}(x) - \sum_{n=1}^{L}\left(\frac{1}{L}\frac{1}{\gamma_n}\sigma^2 r(x,\mathbb{D})\widetilde{\Phi}_n\right)\left(\frac{1}{L}\widetilde{\Phi}_n'\widetilde{Z}(\mathbb{D})\right), \\ &= \widetilde{Z}(x) - \left(\sigma^2 r(x,\mathbb{D})\right)\left(\frac{1}{L^2}\sum_{n=1}^{L}\frac{1}{\gamma_n}\widetilde{\Phi}_n\widetilde{\Phi}_n'\right)\widetilde{Z}(\mathbb{D}), \\ &= \widetilde{Z}(x) - \left(\sigma^2 r(x,\mathbb{D})\right)\left(\sigma^2 r(\mathbb{D},\mathbb{D})\right)^{-1}\widetilde{Z}(\mathbb{D}), \\ &= \widetilde{Z}(x) - \mathbb{E}[\widetilde{Z}(x) \mid \widetilde{Z}(\mathbb{D})], \\ &= \left[\widetilde{Z}(x) \mid \widetilde{Z}(\mathbb{D}) = 0\right]. \end{aligned}$$

The process approximating Z^P *is the conditioned (on a finite set of points) Gaussian process* $Z^{\mathbb{D}}$*. It is a centered Gaussian process of covariance kernel* $\sigma^2\rho^{\mathbb{D}}$ *with:*

$$\rho^{\mathbb{D}}(x,x') \quad = \quad r(x,x') - r(x,\mathbb{D})r(\mathbb{D},\mathbb{D})^{-1}r(\mathbb{D},x') \quad \forall x, x' \in [0,1]^{d_J+d_I}.$$

■

B Proof of proposition 2

This section deals with the proof of proposition 2 which gives a closed form formula of the process Z^P (see equation (5)) for a particular choice of the kernel of $\widetilde{Z}$.

Proof • *Eigenvalue problem:*

The eigenvalue problem can be rewritten as:

$$\begin{array}{rll} & \int_{[0,1]^{d_J}} \left(\sigma^2 r\left((x_J, g(x_J)), (s_J, g(s_J))\right)\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J = \lambda_n \widetilde{\phi}_n(x_J), & \forall x_J \in [0,1]^{d_J}, \\ \Leftrightarrow & \int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) r_I\left(g(x_J) - g(x_J), g(s_J) - g(s_J)\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J = \lambda_n \widetilde{\phi}_n(x_J), & \\ \Leftrightarrow & \int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) \underbrace{r_I(0,0)}_{=1} \widetilde{\phi}_n(s_J) \mathrm{d}s_J = \lambda_n \widetilde{\phi}_n(x_J), & \\ \Leftrightarrow & \int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J = \lambda_n \widetilde{\phi}_n(x_J). & \end{array} \tag{10}$$

- *Expression of ϕ_n:*

  ϕ_n can be rewritten as:

$$\begin{array}{rll} \phi_n(x_J, x_I) & = & \frac{1}{\lambda_n} \int_{[0,1]^{d_J}} \left(\sigma^2 r\left((x_J, x_I), (s_J, g(s_J))\right)\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J, \qquad \forall (x_J, x_I) \in [0,1]^{d_J} \times [0,1]^{d_I}, \\ & = & \frac{1}{\lambda_n} \int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) r_I\left(x_I - g(x_J), g(s_J) - g(s_J)\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J, \\ & = & \frac{1}{\lambda_n} \left(\int_{[0,1]^{d_J}} \left(\sigma^2 r_J(x_J, s_J)\right) \widetilde{\phi}_n(s_J) \mathrm{d}s_J \right) r_I\left(x_I - g(x_J), 0\right)), \\ & = & \frac{1}{\lambda_n} \left(\lambda_n \widetilde{\phi}_n(x_J) \right) r_I\left(x_I - g(x_J), 0\right)), \\ & = & \widetilde{\phi}_n(x_J) r_I\left(x_I - g(x_J), 0\right)). \end{array}$$

  The second last equality is due to the fact that $\widetilde{\phi}_n$ is solution of the eigenvalue problem (10)*.*

- *The P process can be rewritten as:*

$$\begin{array}{rll} Z^P(x_J, x_I) & = & \widetilde{Z}(x_J, x_I) - \sum_{n=1}^{+\infty} \phi_n(x_J, x_I) \int \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(s_J)) \mathrm{d}s_J, \qquad \forall (x_J, x_I) \in [0,1]^{d_J + d_I}, \\ & = & \widetilde{Z}(x_J, x_I) - \sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) r_I\left(x_I - g(x_J), 0\right)) \int \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(s_J)) \mathrm{d}s_J, \\ & = & \widetilde{Z}(x_J, x_I) - \left(\sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) \int \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(s_J)) \mathrm{d}s_J \right) r_I\left(x_I - g(x_J), 0\right)), \end{array}$$

  and, because $\widetilde{Z}(x_J, g(x_J))$ belongs to the sub Gaussian space engendered by the $\{\widetilde{Z}(s_J, g(s_J))\ \forall s_J \in [0,1]^{d_J}\}$, its projection in this subspace is equal to itself:

$$\begin{array}{rll} \widetilde{Z}(x_J, g(x_J)) & = & \mathbb{E}[\widetilde{Z}(x_J, g(x_J)) \mid \widetilde{Z}(s_J, g(s_J)) \forall s_J], \qquad \forall x_J \in [0,1]^{d_J}, \\ & = & \sum_{n=1}^{+\infty} \phi_n(x_J, g(x_J)) \int \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(t_J)) \mathrm{d}s_J, \\ & = & \sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) \underbrace{r_I\left(g(x_J) - g(x_J), 0\right))}_{=1} \int \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(t_J)) \mathrm{d}s_J, \\ & = & \sum_{n=1}^{+\infty} \widetilde{\phi}_n(x_J) \int \widetilde{\phi}_n(s_J) \widetilde{Z}(s_J, g(s_J)) \mathrm{d}s_J. \end{array}$$

  The second equality is the formula of the conditional expectation (see (6)*).*

  So

$$Z^P(x) = \widetilde{Z}(x) - r_I\left(x_I - g(x_J), 0\right) \widetilde{Z}(x_J, g(x_J)).$$

■

C Formulas of the EM procedure

This section gives the formulas of the $\mathcal{Q}_n$ involved in the optimization problems (7) used to estimate the parameters η_n in the EM algorithm. $\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}}$ is the unit column vector of size $nrows\left(\underset{\sim}{\mathbb{X}_1}\right)$. Denoting by $\mathbb{X}^1$ and $\mathbb{X}^2$ two DoE's, $0_{\mathbb{X}^1,\mathbb{X}^2}$ is the null matrix of size $nrows(\mathbb{X}^1) \times nrows(\mathbb{X}^2)$.

Using the expectation of a quadratic form formula to Gaussian vectors:

$$\left\{\begin{array}{lcl} \mathcal{Q}_1(\eta_1,\eta^*) & = & n_{\underset{\sim}{\mathbb{X}_1}} \log\sigma_1^2 + \log\left|\rho_{\theta_1}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right)\right| \\ & & + \frac{Tr\left(\rho_{\theta_1}(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1})^{-1}C_1(\eta^*)\right)}{\sigma_1^2} \\ & & + \frac{(E_1(\eta^*)-m\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}})'\rho_{\theta_1}(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1})^{-1}(E_1(\eta^*)-m\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}})}{\sigma_1^2}, \\ \forall n \in [\![2,N]\!], & & \\ \mathcal{Q}_n(\eta_n,\eta^*) & = & n_{\underset{\sim}{\mathbb{X}_n}} \log\sigma_n^2 + \log\left|\rho_{\theta_n}\left(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n}\right)\right| \\ & & + \frac{Tr\left(\rho_{\theta_n}(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n})^{-1}C_n(\eta^*)\right)}{\sigma_n^2} \\ & & + \frac{E_n(\eta^*)'\rho_{\theta_n}(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n})^{-1}E_n(\eta^*)}{\sigma_n^2}, \end{array}\right.$$

with

$$\left\{\begin{array}{lcll} E_1(\eta^*) & = & m^*\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}} & \\ & & +(\sigma_1^*)^2\rho_{\theta_1^*}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right) Cov_{\eta^*}\left(\mathbf{Y},\mathbf{Y}\right)^{-1}\left(\mathbf{y}-\mathbb{E}_{\eta^*}\left[\mathbf{Y}\right]\right), & \\ C_1(\eta^*) & = & (\sigma_1^*)^2\rho_{\theta_1^*}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right) - (\sigma_1^*)^2\rho_{\theta_1^*}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right) Cov_{\eta^*}\left(\mathbf{Y},\mathbf{Y}\right)^{-1}(\sigma_1^*)^2\rho_{\theta_1^*}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right), & \\ & & & \\ E_n(\eta^*) & = & Cov_{\eta^*}\left(Z_n\left(\underset{\sim}{\mathbb{X}_n}\right),\mathbf{Y}\right) Cov_{\eta^*}\left(\mathbf{Y},\mathbf{Y}\right)^{-1}\left(\mathbf{y}-\mathbb{E}_{\eta^*}\left[\mathbf{Y}\right]\right), & \forall n \in [\![1,N]\!], \\ C_n(\eta^*) & = & (\sigma_n^*)^2\,\rho_{\theta_n^*}\left(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n}\right) & \\ & & -Cov_{\eta^*}\left(Z_n\left(\underset{\sim}{\mathbb{X}_n}\right),\mathbf{Y}\right) Cov_{\eta^*}\left(\mathbf{Y},\mathbf{Y}\right)^{-1} Cov_{\eta^*}\left(\mathbf{Y},Z_n\left(\underset{\sim}{\mathbb{X}_n}\right)\right). & \end{array}\right. \tag{11}$$

Partial analytical solution The optima $m^{(i+1)}$ and $\sigma_n^{(i+1)}$ ($n \in [\![1,N]\!]$) have analytical forms obtained by solving the system formed when the corresponding partial derivatives of the $\mathcal{Q}_n$ vanish. Finally, at each new iteration $i+1$, the goal is to find $\theta_n^{(i+1)}$ ($n \in [\![2,N]\!]$) solution of the following problem

$$\left\{\begin{array}{ll} \min\limits_{\theta_n} & n_{\underset{\sim}{\mathbb{X}_n}} \log\left(\left(\sigma_n^{(i+1)}(\theta_n)\right)^2\right) + \log\left(\left|\rho_{\theta_n}\left(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n}\right)\right|\right), \\ & \\ \text{with } \left(\sigma_n^{(i+1)}(\theta_n)\right)^2 = & \frac{Tr\left(\rho_{\theta_n}(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n})^{-1}C_n(\eta^{(i)})\right)+E_n(\eta^{(i)})'\rho_{\theta_n}(\underset{\sim}{\mathbb{X}_n},\underset{\sim}{\mathbb{X}_n})^{-1}E_n(\eta^{(i)})}{n_{\underset{\sim}{\mathbb{X}_n}}}, \end{array}\right.$$

and $\theta_1^{(i+1)}$ solution of the following problem

$$\left\{\begin{array}{ll} \min\limits_{\theta_1} & n_{\underset{\sim}{\mathbb{X}_1}} \log\left(\left(\sigma_1^{(i+1)}(\theta_1)\right)^2\right) + \log\left(\left|\rho_{\theta_1}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right)\right|\right), \\ & \\ \text{with } \left(\sigma_1^{(i+1)}(\theta_1)\right)^2 = & \frac{Tr\left(\rho_{\theta_1}(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1})^{-1}C_1(\eta^{(i)})\right)+(E_1(\eta^{(i)})-m^{(i+1)}(\theta_1)\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}})'\rho_{\theta_1}(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1})^{-1}(E_1(\eta^{(i)})-m^{(i+1)}(\theta_1)\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}})}{n_{\underset{\sim}{\mathbb{X}_1}}}, \\ & \\ \text{and } m^{(i+1)}(\theta_1) = & \frac{\mathbf{1}'_{\underset{\sim}{\mathbb{X}_1}}\rho_{\theta_1}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right)^{-1}E_1(\eta^{(i)})}{\mathbf{1}'_{\underset{\sim}{\mathbb{X}_1}}\rho_{\theta_1}\left(\underset{\sim}{\mathbb{X}_1},\underset{\sim}{\mathbb{X}_1}\right)^{-1}\mathbf{1}_{\underset{\sim}{\mathbb{X}_1}}}, \end{array}\right.$$

where $E_n(\eta^{(i)})$ and $C_n(\eta^{(i)})$ ($n \in [\![1, N]\!]$) are given in equation (11).

D Proof of the monotonicity of the sequence from the EM algorithm

Let $\mathbf{Y} = (Y_1(\mathbb{X}_1), \cdots, Y_N(\mathbb{X}_N))$ denote the observed data. Let $\mathbf{U} = \left(Z_1\left(\widetilde{\mathbb{X}}_1 \backslash \mathbb{X}_1\right), Z_2\left(\widetilde{\mathbb{X}}_2 \backslash \mathbb{X}_2\right), \ldots, Z_{N-1}\left(\widetilde{\mathbb{X}}_{N-1} \backslash \mathbb{X}_{N-1}\right)\right)$ denote the unknown data. The complete data is equal to: $\mathbf{T} = \left(Z_1\left(\widetilde{\mathbb{X}}_1\right), \cdots, Z_{N-1}\left(\widetilde{\mathbb{X}}_{N-1}\right), Z_N(\mathbb{X}_N)\right)$. Let $\mathbf{y}$, $\mathbf{u}$, and $\mathbf{t}$ be the realizations of respectively $\mathbf{Y}$, $\mathbf{U}$, and $\mathbf{T}$. By definition of the conditional density:

$$h_{\mathbf{Y};\eta}(\mathbf{y}) = \frac{h_{\mathbf{T};\eta}(t)}{\underbrace{h_{\mathbf{U} \mid \mathbf{Y}=\mathbf{y};\eta}(\mathbf{u})}_{\widetilde{\mathbf{U}}}}.$$

So

$$\mathcal{L}(\eta; \mathbf{y}) = \frac{\mathcal{L}_c(\eta; t)}{h_{\widetilde{\mathbf{U}};\eta}(\mathbf{u})}.$$

So (up to a constant)

$$l(\eta; \mathbf{y}) \stackrel{c}{=} l_c(\eta; \mathbf{t}) + 2\log(h_{\widetilde{\mathbf{U}};\eta}(\mathbf{u})).$$

Replacing the observations by the corresponding random variables

$$l(\eta; \mathbf{Y}) \stackrel{c}{=} l_c(\eta; \mathbf{T}) + 2\log(h_{\widetilde{\mathbf{U}};\eta}(\widetilde{\mathbf{U}})).$$

Taking the expectation assuming η' and conditioning by $\mathbf{Y} = \mathbf{y}$

$$l(\eta; \mathbf{y}) \stackrel{c}{=} \mathcal{Q}(\eta, \eta') + 2\mathcal{R}(\eta, \eta'),$$

with

$$\mathcal{R}(\eta, \eta') = \mathbb{E}_{\eta'}\left[\log(h_{\widetilde{\mathbf{U}};\eta}(\widetilde{\mathbf{U}})) \mid \mathbf{Y} = \mathbf{y}\right].$$

The difference between the loss function taken at the updated and previous terms of the EM sequence is the sum of two quantities.

$$l(\eta^{(i+1)}) - l(\eta^{(i)}) = \underbrace{\mathcal{Q}(\eta^{(i+1)}, \eta^{(i)}) - \mathcal{Q}(\eta^{(i)}, \eta^{(i)})}_{\leqslant 0} + 2\underbrace{\left(\mathcal{R}(\eta^{(i+1)}, \eta^{(i)}) - \mathcal{R}(\eta^{(i)}, \eta^{(i)})\right)}_{=R}.$$

The first quantity is negative by definition of the EM algorithm. The second quantity can be rewritten as:

$$\begin{aligned}
R &= \mathcal{R}(\eta^{(i+1)}, \eta^{(i)}) - \mathcal{R}(\eta^{(i)}, \eta^{(i)}), \\
&= \mathbb{E}_{\eta^{(i)}}\left[\log\left(h_{\widetilde{\mathbf{U}};\eta^{(i+1)}}(\widetilde{\mathbf{U}})\right)\right] - \mathbb{E}_{\eta^{(i)}}\left[\log\left(h_{\widetilde{\mathbf{U}};\eta^{(i)}}(\widetilde{\mathbf{U}})\right)\right], \\
&= \mathbb{E}_{\eta^{(i)}}\left[\log\left(\frac{h_{\widetilde{\mathbf{U}};\eta^{(i+1)}}(\widetilde{\mathbf{U}})}{h_{\widetilde{\mathbf{U}};\eta^{(i)}}(\widetilde{\mathbf{U}})}\right)\right], \\
&\leqslant \log\left(\mathbb{E}_{\eta^{(i)}}\left[\frac{h_{\widetilde{\mathbf{U}};\eta^{(i+1)}}(\widetilde{\mathbf{U}})}{h_{\widetilde{\mathbf{U}};\eta^{(i)}}(\widetilde{\mathbf{U}})}\right]\right), \\
&\leqslant \log\left(\int \frac{h_{\widetilde{\mathbf{U}};\eta^{(i+1)}}(\mathbf{u})}{h_{\widetilde{\mathbf{U}};\eta^{(i)}}(\mathbf{u})} h_{\widetilde{\mathbf{U}};\eta^{(i)}}(\mathbf{u})\mathrm{d}\mathbf{u}\right), \\
&\leqslant \log\left(\int h_{\widetilde{\mathbf{U}};\eta^{(i+1)}}(\mathbf{u})\mathrm{d}\mathbf{u}\right), \\
&\leqslant \log(1), \\
&\leqslant 0.
\end{aligned}$$

Indeed the inequality $l(\eta^{(i+1)}) - l(\eta^{(i)}) \leqslant 0$ is verified by the sequence $\left(\eta^{(i)}\right)_i$ built following the EM algorithm.

References

[Auder et al., 2012] Auder, B., De Crecy, A., Iooss, B., and Marques, M. (2012). Screening and metamodeling of computer experiments with functional outputs. application to thermal–hydraulic computations. Reliability Engineering & System Safety, 107:122–131.

[Bachoc et al., 2020] Bachoc, F., Lopera, A. F. L., and Roustant, O. (2020). Sequential construction and dimension reduction of gaussian processes under inequality constraints. arXiv preprint arXiv:2009.04188.

[Cheng and Titterington, 1994] Cheng, B. and Titterington, D. M. (1994). Neural networks: A review from a statistical perspective. Statistical science, pages 2–30.

[Clarke et al., 2005] Clarke, S. M., Griebsch, J. H., and Simpson, T. W. (2005). Analysis of support vector regression for approximation of complex engineering analyses.

[Da Veiga and Marrel, 2012] Da Veiga, S. and Marrel, A. (2012). Gaussian process modeling with inequality constraints. In Annales de la Faculté des sciences de Toulouse: Mathématiques, volume 21, pages 529–555.

[Dupuy et al., 2015] Dupuy, D., Helbert, C., Franco, J., et al. (2015). Dicedesign and diceeval: Two r packages for design and analysis of computer experiments. Journal of Statistical Software, 65(11):1–38.

[Forrester et al., 2008] Forrester, A., Sobester, A., and Keane, A. (2008). Engineering design via surrogate modelling: a practical guide. John Wiley & Sons.

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001). The elements of statistical learning, volume 1. Springer series in statistics New York.

[Gauthier and Bay, 2012] Gauthier, B. and Bay, X. (2012). Spectral approach for kernel-based interpolation. In Annales de la Faculté des sciences de Toulouse: Mathématiques, volume 21, pages 439–479.

[Hebbal et al., 2021] Hebbal, A., Brevault, L., Balesdent, M., Talbi, E.-G., and Melab, N. (2021). Multi-fidelity modeling with different input domain definitions using deep gaussian processes. Structural and Multidisciplinary Optimization, pages 1–22.

[Iooss and Prieur, 2019] Iooss, B. and Prieur, C. (2019). Shapley effects for sensitivity analysis with correlated inputs: comparisons with sobol'indices, numerical estimation and applications. International Journal for Uncertainty Quantification, 9(5).

[Jones et al., 1998] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13(4):455–492.

[Kennedy and O'Hagan, 2000] Kennedy, M. C. and O'Hagan, A. (2000). Predicting the output from a complex computer code when fast approximations are available. Biometrika, 87(1):1–13.

[Le Gratiet and Garnier, 2014] Le Gratiet, L. and Garnier, J. (2014). Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. International Journal for Uncertainty Quantification, 4(5).

[Lefebvre et al., 2010] Lefebvre, S., Roblin, A., Varet, S., and Durand, G. (2010). A methodological approach for statistical evaluation of aircraft infrared signature. Reliability Engineering & System Safety, 95(5):484–493.

[Lopera, 2019] Lopera, A. F. L. (2019). Gaussian Process Modelling under Inequality Constraints. PhD thesis, Université de Lyon.

[Maatouk and Bay, 2017] Maatouk, H. and Bay, X. (2017). Gaussian process emulators for computer experiments with inequality constraints. Mathematical Geosciences, 49(5):557–582.

[Makowski et al., 2006] Makowski, D., Naud, C., Jeuffroy, M.-H., Barbottin, A., and Monod, H. (2006). Global sensitivity analysis for calculating the contribution of genetic parameters to the variance of crop model prediction. Reliability Engineering & System Safety, 91(10-11):1142–1147.

[Oakley and O'Hagan, 2004] Oakley, J. E. and O'Hagan, A. (2004). Probabilistic sensitivity analysis of complex models: a bayesian approach. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 66(3):751–769.

[Pelamatti et al., 2021] Pelamatti, J., Brevault, L., Balesdent, M., Talbi, E.-G., and Guerin, Y. (2021). Bayesian optimization of variable-size design space problems. Optimization and Engineering, 22(1):387–447.

[Pronzato and Müller, 2012] Pronzato, L. and Müller, W. G. (2012). Design of computer experiments: space filling and beyond. Statistics and Computing, 22(3):681–701.

[Roustant et al., 2012] Roustant, O., Ginsbourger, D., and Deville, Y. (2012). Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization.

[Sacks et al., 1989] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. Statistical science, pages 409–423.

[Saltelli, 2000] Saltelli, A. (2000). Sensitivity analysis, edited by: Saltelli, a., chan, k., and scott, em.

[Santner et al., 2003] Santner, T. J., Williams, B. J., Notz, W., and Williams, B. J. (2003). The design and analysis of computer experiments, volume 1. Springer.

[Sobol, 1993] Sobol, I. M. (1993). Sensitivity analysis for non-linear mathematical models. Mathematical modelling and computational experiment, 1:407–414.

[Valeo,] Valeo. Internal documentation.

[Williams and Rasmussen, 2006] Williams, C. K. and Rasmussen, C. E. (2006). Gaussian processes for machine learning, volume 2. MIT press Cambridge, MA.

Bibliography

[Bachoc et al., 2020] Bachoc, F., Lopera, A. F. L., and Roustant, O. (2020). Sequential construction and dimension reduction of gaussian processes under inequality constraints. *arXiv preprint arXiv:2009.04188*.

[Cheng and Titterington, 1994] Cheng, B. and Titterington, D. M. (1994). Neural networks: A review from a statistical perspective. *Statistical science*, pages 2–30.

[Clarke et al., 2005] Clarke, S. M., Griebsch, J. H., and Simpson, T. W. (2005). Analysis of support vector regression for approximation of complex engineering analyses. *Journal of Mechanical Design*.

[Da Veiga et al., 2021] Da Veiga, S., Gamboa, F., Iooss, B., and Prieur, C. (2021). *Basics and Trends in Sensitivity Analysis: Theory and Practice in R*. SIAM.

[Da Veiga and Marrel, 2012] Da Veiga, S. and Marrel, A. (2012). Gaussian process modeling with inequality constraints. In *Annales de la Faculté des sciences de Toulouse: Mathématiques*, volume 21, pages 529–555.

[Dupuy et al., 2015] Dupuy, D., Helbert, C., Franco, J., et al. (2015). Dicedesign and diceeval: Two r packages for design and analysis of computer experiments. *Journal of Statistical Software*, 65(11):1–38.

[Forrester et al., 2008] Forrester, A., Sobester, A., and Keane, A. (2008). *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons.

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

[Gauthier, 2011] Gauthier, B. (2011). *Approche spectrale pour l'interpolation à noyaux et positivité conditionnelle*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne.

[Gauthier and Bay, 2012a] Gauthier, B. and Bay, X. (2012a). Spectral approach for kernel-based interpolation. In *Annales de la Faculté des sciences de Toulouse: Mathématiques*, volume 21, pages 458–460.

[Gauthier and Bay, 2012b] Gauthier, B. and Bay, X. (2012b). Spectral approach for kernel-based interpolation. In *Annales de la Faculté des sciences de Toulouse: Mathématiques*, volume 21, pages 439–479.

[Gonon et al., 2021] Gonon, T., Helbert, C., Blanchet-Scalliet, C., and Demory, B. (2021). Gaussian process regression on nested spaces. *HAL*.

[Gourdon, 1994] Gourdon, X. (1994). *Les maths en tête*. Ellipses-Marketing.

[Hebbal et al., 2021] Hebbal, A., Brevault, L., Balesdent, M., Talbi, E.-G., and Melab, N. (2021). Multi-fidelity modeling with different input domain definitions using deep gaussian processes. *Structural and Multidisciplinary Optimization*, pages 1–22.

[Henner et al., 2019] Henner, M., Demory, B., Gonon, T., and Helbert, C. (2019). Sampling strategies for metamodel enrichment and automotive fan optimization. In *European Conference on Turbomachinery Fluid Dynamics and Thermodynamics*.

[Iooss and Prieur, 2019] Iooss, B. and Prieur, C. (2019). Shapley effects for sensitivity analysis with correlated inputs: comparisons with sobol indices, numerical estimation and applications. *International Journal for Uncertainty Quantification*, 9(5).

[Jones et al., 1998] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

[Kennedy and O'Hagan, 2000] Kennedy, M. C. and O'Hagan, A. (2000). Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13.

[Le Gratiet, 2013a] Le Gratiet, L. (2013a). *Multi-fidelity Gaussian process regression for computer experiments*. PhD thesis, Université Paris-Diderot. page 182.

[Le Gratiet, 2013b] Le Gratiet, L. (2013b). *Multi-fidelity Gaussian process regression for computer experiments*. PhD thesis, Université Paris-Diderot.

[Le Gratiet and Garnier, 2014] Le Gratiet, L. and Garnier, J. (2014). Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. *International Journal for Uncertainty Quantification*, 4(5).

[Lopera, 2019] Lopera, A. F. L. (2019). *Gaussian Process Modelling under Inequality Constraints*. PhD thesis, Ecole des Mines de Saint-Etienne.

[López-Lopera et al., 2017] López-Lopera, A. F., Bachoc, F., Durrande, N., and Roustant, O. (2017). Finite-dimensional gaussian approximation with linear inequality constraints. *arXiv preprint arXiv:1710.07453*.

[Maatouk and Bay, 2017] Maatouk, H. and Bay, X. (2017). Gaussian process emulators for computer experiments with inequality constraints. *Mathematical Geosciences*, 49(5):557–582.

[Morris and Mitchell, 1995] Morris, M. D. and Mitchell, T. J. (1995). Exploratory designs for computational experiments. *Journal of statistical planning and inference*, 43(3):381–402.

[Oakley and O'Hagan, 2004] Oakley, J. E. and O'Hagan, A. (2004). Probabilistic sensitivity analysis of complex models: a bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769.

[Pelamatti et al., 2021] Pelamatti, J., Brevault, L., Balesdent, M., Talbi, E.-G., and Guerin, Y. (2021). Bayesian optimization of variable-size design space problems. *Optimization and Engineering*, 22(1):387–447.

[Picheny et al., 2010] Picheny, V., Ginsbourger, D., Roustant, O., Haftka, R. T., and Kim, N.-H. (2010). Adaptive designs of experiments for accurate approximation of a target region. *Journal of Mechanical Design*.

[Pronzato and Müller, 2012] Pronzato, L. and Müller, W. G. (2012). Design of computer experiments: space filling and beyond. *Statistics and Computing*, 22(3):681–701.

[Roustant et al., 2012] Roustant, O., Ginsbourger, D., and Deville, Y. (2012). Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of statistical software*, 51:1–55.

[Sacks et al., 1989] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical science*, pages 409–423.

[Saltelli et al., 2008] Saltelli, A., Chan, K., and Scott, E. (2008). *Sensitivity Analysis*. John Wiley & Sons.

[Santner et al., 2003a] Santner, T. J., Williams, B. J., Notz, W., and Williams, B. J. (2003a). *The design and analysis of computer experiments*, volume 1. Springer.

[Santner et al., 2003b] Santner, T. J., Williams, B. J., Notz, W., and Williams, B. J. (2003b). *The design and analysis of computer experiments*, volume 1, pages 145–245. Springer.

[Sobol, 1993] Sobol, I. M. (1993). Sensitivity analysis for non-linear mathematical models. *Mathematical modelling and computational experiment*, 1:407–414.

[Valeo,] Valeo. Internal documentation.

[Williams and Rasmussen, 2006] Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.

[Zertuche, 2015] Zertuche, F. (2015). *Assessment of uncertainty in computer experiments when working with multifidelity simulators.* PhD thesis, Université Grenoble Alpes.

dernière page de la thèse

AUTORISATION DE SOUTENANCE

Vu les dispositions de l'arrêté du 25 mai 2016,

Vu la demande du directeur de thèse

Mesdames C. BLANCHET-SCALLIET et C. HELBERT

et les rapports de

M. L. PRONZATO
Directeur de Recherche CNRS - CNRS Laboratoire I3S - UMR 7271 - Université Côte d'Azur - Les Algorithmes - Bât. Euclide B - 2000 route des Lucioles - BP 121
06903 Sophia Antipolis cedex

et de

M. B. IOOS
Docteur HDR - Senior Researcher and Project Manager - EDF R&D - 6 quai Watier
78401 Chatou cedex

Monsieur GONON Thierry

est autorisé à soutenir une thèse pour l'obtention du grade de **DOCTEUR**

Ecole doctorale Mathématiques et Informatique

Fait à Ecully, le 1er mars 2022

Pour le directeur de l'Ecole centrale de Lyon
Le directeur des Formations

Grégory VIAL