Assessment of uncertainty in computer experiments when working with multifidelity simulators.

Federico Zertuche

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques Appliquées**

Arrêté ministérial : 7 août 2006

Présentée par

Federico Zertuche

Thèse dirigée par **Anestis Antoniadis**
et codirigée par **Céline Helbert**

préparée au sein **du Laboratoire Jean Kuntzmann**
et de **l'Ecole Doctorale ED-MSTII**

Utilisation de simulateurs multi-fidélité pour les études d'incertitudes dans les codes de calcul

Thèse soutenue publiquement le 08/10/2015 ,
devant le jury composé de :

M. Josselin Garnier
Professeur, Université Paris Diderot, Rapporteur

M. Alberto Pasanisi
Project Manager, EDF - European Institute For Energy Reseach (EIFER), Rapporteur

M. Fabrice Gamboa
Professeur, Institut de Mathématiques de Toulouse, Président

M. Mathieu Couplet
Research engineer, EDF, Chatou, Examinateur

Mme Clementine Prieur
Professeur, Université Joseph Fourier, Examinateur

M. Anestis Antoniadis
Professeur, Université Joseph Fourier, Directeur de thèse

Mme Céline Helbert
Maître de Conférences, Institut Camille Jordan, Co-Directeur de thèse

Résumé

Les simulations par ordinateur sont un outil de grande importance pour les mathématiciens appliqués et les ingénieurs. Elles sont devenues plus précises mais aussi plus compliquées. Tellement compliquées, que le temps de lancement par calcul est prohibitif. Donc, plusieurs aspects de ces simulations sont mal compris. Par exemple, souvent ces simulations dépendent des paramètres qu'ont une valeur inconnue.

Un metamodèle est une reconstruction de la simulation. Il produit des réponses proches à celles de la simulation avec un temps de calcul très réduit. Avec ce metamodèle il est possible d'étudier certains aspects de la simulation. Il est construit avec peu de données et son objectif est de remplacer la simulation originale.

Ce travail est concerné avec la construction des metamodèles dans un cadre particulier appelé multi-fidélité. En multi-fidélité, le metamodèle est construit à partir des données produites par une simulation objective et des données qu'ont une relation avec cette simulation. Ces données approximées peuvent être générés par des versions dégradées de la simulation ; par des anciennes versions qu'ont été largement étudiées ou par une autre simulation dans laquelle une partie de la description est simplifiée.

En apprenant la différence entre les données il est possible d'incorporer l'information approximée et ce ci peut nous conduire vers un metamodèle amélioré. Deux approches pour atteindre ce but sont décrites dans ce manuscrit : la première est basée sur des modèles avec des processus

gaussiens et la seconde sur une décomposition à base d'ondelettes. La première montre qu'en estimant la relation il est possible d'incorporer des données qui n'ont pas de valeur autrement. Dans la seconde, les données sont ajoutées de façon adaptative pour améliorer le metamodèle.

L'objet de ce travail est d'améliorer notre compréhension sur comment incorporer des données approximées pour produire des metamodèles plus précis. Travailler avec un metamodèle multi-fidélité nous aide à comprendre en détail ces éléments. A la fin une image globale des parties qui forment ce metamodèle commence à s'esquisser : les relations et différences entres les données deviennent plus claires.

Abstract

A very important tool used by applied mathematicians and engineers to model the behavior of a system are computer simulations. They have become increasingly more precise but also more complicated. So much, that they are very slow to produce an output and thus difficult to sample so that many aspects of these simulations are not very well understood. For example, in many cases they depend on parameters whose value is unknown.

A metamodel is a reconstruction of the simulation. It requires much less time to produce an output that is close to what the simulation would. By using it, some aspects of the original simulation can be studied. It is built with very few samples and its purpose is to replace the simulation.

This thesis is concerned with the construction of a metamodel in a particular context called multi-fidelity. In multi-fidelity the metamodel is constructed using the data from the target simulation along other samples that are related. These approximate samples can come from a degraded version of the simulation; an old version that has been studied extensively or a another simulation in which a part of the description is simplified.

By learning the difference between the samples it is possible to incorporate the information of the approximate data and this may lead to an enhanced metamodel. In this manuscript two approaches that do this are studied: one based on Gaussian process modeling and another based on a coarse to fine Wavelet decomposition. The fist method shows how by

estimating the relationship between two data sets it is possible to incorporate data that would be useless otherwise. In the second method an adaptive procedure to add data systematically to enhance the metamodel is proposed.

The object of this work is to better our comprehension of how to incorporate approximate data to enhance a metamodel. Working with a multi-fidelity metamodel helps us to understand in detail the data that nourish it. At the end a global picture of the elements that compose it is formed: the relationship and the differences between all the data sets become clearer.

Contents

Chapter 1

Introduction

1.1 Computer experiments

Computer simulations have become the main tool used by applied mathematicians to study many different phenomena. They are used to model very large and complex systems like the ocean, a flying plane or parts of nuclear power plant. These models are increasingly more precise but also more complex to program, to sample and to understand. Here we try to synthesis data from such simulations.

In our case these simulations take a multidimensional input that contains all the parameters that describe the object of study and produce one or many responses that are function of the parameters.

More often than not, many aspects of the simulation are not very well understood. This includes the real value of the parameters and the influence they have on the output. Researchers and engineers use statistical techniques like Sensitivity Analysis to try to address these problems. In practice this means that the model has to be sampled over a very big set of inputs and this is unfeasible in many cases due to time constraints. For example, some simulations can take up to several days to produce a single response and some statistical techniques require thousands of samples.

One possible solution is to use very few samples, like a hundred, to build a fast metamodel that takes in the same input and produces a response that is "close" to what the simulation would. Then, use this metamodel to study the statistical properties we are interested in. This is the alternative explored in the thesis.

1.2 Prediction

The way this problem is formalized is by thinking about the computer simulation we try to approximate as a function f of a vector of parameters x.

If we note $y = f(x)$ the target output, then we sample the function over a few inputs to get a data set $(x_1, y_1), ..., (x_n, y_n)$. The problem is to look for a function $\widehat{y}(x)$ built using the observed values and that predicts f at an unobserved input x.

This function $\widehat{y}$ is called a predictor and it is usually chosen from a family of models $\mathcal{Y}$ as the one that produces the outputs that are the closest to the data. The family of models can be defined by some parameters or not or it can be a set of functions with some regularity properties. Normally, we choose the predictor by solving a problem like

$$\underset{\widehat{y}\in\mathcal{Y}}{\text{minimize}} \quad \sum_{i=1}^{n} \left(\widehat{y}(x_i) - y_i\right)^2 .$$

In this first approach we do not take into account the fact that we are uncertain about the chosen model. For example, it could happen that the optimal predictor is very close to the data but when the inputs are altered by a plausible amount, the predicted value changes a lot. The methods studied in the manuscript are used in this context.

Two common ways of taking into account this uncertainty are to use a stochastic model in which we assume that the outputs generated by the

simulation are random variables and solve

$$\underset{\widehat{Y}\in\mathcal{Y}}{\text{minimize}} \quad \sum_{i=1}^{n} E\left(\widehat{Y}(x_i) - Y_i\right)^2$$

or to consider a worst-case analysis indexed by a set $\mathcal{H}$ in which we solve

$$\underset{\widehat{y}\in\mathcal{Y}}{\text{minimize}} \quad \sup_{h\in\mathcal{H}} \sum_{i=1}^{n} \left(\widehat{y}(x_i; h) - y_i\right)^2.$$

Two different prediction techniques are studied. Each one uses one of these two paradigms in a particular context called multi-fidelity. The first one is based on Gaussian processes models and the second on wavelets.

1.3 Multi-fidelity

Usually more data used to build a predictor translates into a better prediction. Multi-fidelity is about using different data sets that are related to do such a task.

For example, we can think of simulations in which the precision of the output can be tuned. This happens when we solve a partial differential equation numerically. When we do so, we compute an approximate solution over a grid. This numerical approximation is the computer simulation. The precision of the output depends on the size of the grid: a finer grid produces a more precise response. In this case, the key is that the time it takes to produce the response increases with the precision. It can happen that it is better to have a small fine grid data set along a large data set of coarser observations in which we explore the parameter space in detail instead of only having a slightly bigger fine grid one.

Another example is something that is very usual in research and development departments. Here we want to understand a new version of a simulator. Normally, there are old versions of the same simulator that have

been studied in detail and that could be much faster. We could use this additional data to build an enhanced predictor.

Finally, we can think of using models in which we simplify a part of the description. In fluid dynamics sometimes two dimensional discretizations of a three dimensional model are available. Also, removing some physical properties is a common practice. There are even different ribosomes that read genes with more or less precision. In any of these cases we could use all the data sets available.

In all the examples above, there is always a target simulation and the other data is produced by what we consider as approximations. In particular, we will always assume that the data sets produced by these approximations are ordered according to some fidelity parameter.

For the stochastic approach, we follow Kennedy and O'Hagan [30], Peter Quian [43] and Loic Le Gratiet [33]. They propose a model based on Gaussian processes in which the relationship between two successive levels is parametric. Our first objective is to extend this model to more general situations where the relationship between two successive levels of fidelity cannot be modeled parametrically.

For the second method, we transpose some results of Daniel Castano [8] based on a coarse to fine wavelet approximations to our multi-fidelity problem. Here we do not make any explicit assumption about the relationship between any of the levels. Instead we will judge how close are the two functions are by looking at the difference between their decomposition coefficients. Our second objective is to use this type of decomposition to allocate observations sequentially. This second method can be considered as an alternative to the Gaussian processes regression.

1.4 Outline of the manuscript

In chapter 2 we introduce formally the prediction problem under the Gaussian process hypothesis. We study the parametric form of the mean and covariance functions that define the process and some of the techniques used to estimate them. Then, we define the optimal predictor in this case and describe the existing methods of multi-fidelity prediction.

The second part of chapter 2 is dedicated to the first extensions of the usual multi-fidelity method based on Gaussian processes. First, we study an EM algorithm to estimate the parameters of the model for the case in which the observations are made over disjoint sets. Then, we extend the multi-fidelity method for a polynomial relationship.

In chapter 3 we propose a model in which the relationship between two consecutive levels of fidelity is unknown. We estimate this relationship using locally linear polynomials. We study how this approach can improve the prediction and we compare it to existing multi-fidelity models in several examples.

Chapter 4 is dedicated to the case study that motivated this work. It is also an occasion to apply the methods described above to a practical case. The analyzed simulator is related to the study of a thermal shock inside the cooling system of a nuclear power plant. We also present a new algorithm to build nested designs for the multi-fidelity regression models.

Chapter 5 is devoted to our second method. We start by describing a coarse to fine wavelet based algorithm. A multi-fidelity adaptation of this algorithm in which we use the one data set algorithm to build an adaptive sampling scheme is proposed.

In the last chapter 6 we make some final remarks and discuss possible research directions regarding the problems studied.

Chapter 2

Gaussian processes and Multi-fidelity

In this chapter we deal with optimal prediction for least squares when the data we observe are supposed to be the outcomes of Gaussian random variables. In order to make a formal description of the techniques we define what a Gaussian process is.

Then, we describe how Gaussian processes are used to model functions through a parametric form for the mean and covariance of the process. We briefly describe how to make inference on the parameters.

In the second part of the chapter we introduce the multi-fidelity regression problem and we describe a model that was proposed first by by Marc Kennedy and Tony O'Hagan [30], then extended by Peter Quian [43] and Loic Le Gratiet [35] who proposed a sequential prediction.

We present original proofs regrading the sequential prediction and this will permit us to propose a first extension of the model using a polynomial relationship. This model retains the sequential prediction property.

We extend the model in a second direction by relaxing a crucial requirement for sequential prediction, that is nested observations. We propose an EM algorithm to deal with the case of observations made over disjoint sets.

2.1 Prediction and Gaussian processes

A general framework of prediction starts with a series of observations of the form $Y_{x_1}, ..., Y_{x_n}$ where $x \in \boldsymbol{R}^d$ is called the input or explanatory variable and $Y_x \in \boldsymbol{R}$ is the output or quantity of interest.

We assume that $Y_x = m(x)$ where m is an unknown function and the problem consists in estimating m by using the observed data to make predictions of the output at some unobserved values.

If we note $\mathcal{X}$ the set of inputs; $\mathcal{O}$ the set of all the observable variables and $\mathcal{Y}$ the set of the outputs, then any measurable function of the inputs $t(x; O)$ built using a subset $O \in \mathcal{O}$ is called a predictor. For example

$$t(x; Y_{x_1}, ..., Y_{x_n}) = \frac{1}{n} \sum_{j=1}^{n} Y_{x_j}$$

is a predictor of $m(x)$ that is equal to the mean of the observed values for any x.

To choose a predictor we need to measure how effective they are. In our case we measure the effectiveness of the prediction by computing the mean squared error $E[|Y_x - t(x; O)|^2]$ and we define the optimal prediction as the function that minimizes this quantity.

If y is any predicted value at x, then its mean square error is equal to $EY_x^2 - 2yEY_x + y^2$ when $EY_x^2 < \infty$. From this expression we can deduce that the value that minimizes the mean square error is given by EY_x.

If we observe $Y_n := [Y_{x_1}, ..., Y_{x_n}]^T$, then the best possible value we can produce using the data is the conditional expectation $\tilde{Y}_x := E[Y_x|Y_n]$. A proof of this fact can be found in for example, [54].

In general $\tilde{Y}_x$ is difficult to compute if we do not know the joint distribution of the vector $[Y_x, Y_{x_1}, ..., Y_{x_n}]^T$. One way to avoid this difficulty

is to look for the best linear approximation in Y_n instead. If we write $l(x, Y_n) := \sum_{j=1}^{n} \beta_{j,x} Y_{x_j}$ where $\beta_x^n = [\beta_{1,x}, ..., \beta_{n,x}]^T$ is a vector of unknown coefficients that depend on x, then

$$\frac{\partial}{\partial \beta_{k,x}} E[|Y_x - l(x; Y_n)|^2] = 0 \Leftrightarrow E[(Y_x - l(x; Y_n))Y_{x_k}] = 0 \quad , k = 1, ..., n.$$

For the time being, lets assume that all the random variables have 0 mean. Since $E[(Y_x - \widehat{Y}_x)Y_{x_k}] = 0$ we can write

$$\sum_{j=1}^{n} \beta_{j,x} E[Y_{x_j} Y_{x_k}] = E(Y_{x_k} Y_x)$$

for every $k = 1, ..., n$. Using a matrix notation this system of linear equations is equivalent to $\Gamma(Y_n)\beta_x^n = \Gamma(Y_n, Y_x)$. We note $\Gamma(Y_n)$ the covariance matrix of the observations; $\Lambda(Y_n)$ its inverse and $\Gamma(Y_n, Y_x)$ the vector with coordinates $E(Y_{x_k} Y_x)$. If the covariance matrix is invertible, then we can compute the optimal set of coefficients $\widehat{\beta_x^n}$ as $\Lambda(Y_n)\Gamma(Y_n, Y_x)$. The optimal linear predictor is $\widehat{Y}_x = Y_n^T \widehat{\beta_x^n}$. We can compute its prediction error $E[|Y_x - \widehat{Y}_x|^2]$ by noticing that

$$\begin{aligned} E[|Y_x - \widehat{Y}_x|^2] &= EY_x^2 - 2EY_x\widehat{Y}_x + E\widehat{Y}_x^2 \\ &= EY_x^2 - E\widehat{Y}_x^2 \\ &= EY_x^2 - \Gamma(Y_n, Y_x)^T \Lambda(Y_n)\Gamma(Y_n, Y_x) \end{aligned}$$

since

$$\begin{aligned} EY_x\widehat{Y}_x &= \Gamma(Y_n, Y_x)^T \Lambda(Y_n)\Gamma(Y_n, Y_x) \\ &= \widehat{\beta_x^n}^T \Lambda(Y_n)\widehat{\beta_x^n} \\ &= E\widehat{Y}_x^2. \end{aligned}$$

By looking at the prediction and prediction error formulas we can see that all we need to compute the two is the mean of Y_x^2, the covariance matrix $\Gamma(Y_n)$, its inverse $\Lambda(Y_n)$ and the vector $\Gamma(Y_n, Y_x)$.

In general linear predictors do worse than non-linear ones. This means

$E[|Y_x - \widehat{Y}_x|^2] \geq E[|Y_x - t(x, O)|^2]$. However, when the joint distribution of the vector $[Y_x, Y_{x_1}, ..., Y_{x_n}]^T$ is Gaussian the optimal linear predictor is the optimal predictor.

To explain this we need to introduce some concepts and to think about the problem in a somewhat different way. It is important to recall that we assumed that

$$E[Y_x] = m(x).$$

and that we want to estimate the unknown function m.

We argued, but did not show, that when the joint distribution of the vector $[Y_x, Y_{x_1}, ..., Y_{x_n}]^T$ is Gaussian, the linear predictor is optimal. If we want to make a prediction at another point, say x' we would need $[Y_{x'}, Y_{x_1}, ..., Y_{x_n}]^T$ to be Gaussian too. The same is true if we have a different set of observations. If we want the linear predictor to be optimal we need any finite vector $[Y_x, Y_{x_1}, ..., Y_{x_n}]^T$ to be Gaussian.

To have a coherent predictor we also need that if we have two vectors with some common coordinates like

$$[Y_x, Y_{x_1}, Y_{x_2}, Y_{x_3}]^T$$
$$[Y_x, Y_{x_1}, Y_{x_2}]^T$$

then, the second vector seen as a subset of the first maintains its distribution. In this case, the same mean and covariance.

The two conditions above are equivalent to say that we assume a priori that Y_x is a Gaussian process. This is a prior on a space of functions. Sometimes this type of techniques are called nonparametric Bayesian and there are some similar ideas for problems of classification where Dirichlet processes are used.

For the definitions of Gaussian vectors and processes you can look at

appendix A. We write

$$Y_x \sim GP(\mu(x), \Gamma(Y_x, Y_{x'}))$$

to say that Y_x is a Gaussian process with mean $\mu(x) = EY_x$ and covariance $Cov(Y_x, Y_{x'}) = \Gamma(Y_x, Y_{x'})$.

If we use the formulas we already derived for the optimal linear predictor, we have that given $Y_n := [Y_{x_1}, ..., Y_{x_n}]^T$ the predictor and its error are

$$\widehat{Y}_x := E[Y_x|Y_n] = \mu(x) + \Gamma(Y_n, Y_x)\Lambda(Y_n)(Y_n - E[Y_n]) \tag{2.1}$$

$$\widehat{\Gamma}(x, x) := E[|Y_x - \widehat{Y}_x|^2|Y_n] = \Gamma(Y_x, Y_x) - \Gamma(Y_n, Y_x)\Lambda(Y_n)\Gamma(Y_x, Y_n). \tag{2.2}$$

In fact by using properties of Gaussian vectors, we have the whole conditional distribution

$$[Y_x|Y_n] \sim N(\widehat{Y}_x, \widehat{\Gamma}(x, x')).$$

The predictor in (2.1) is a function of the index x and the vector of observations Y_n. It is the sum of the mean of the process and a special weighted sum of the residuals meaning that it adapts to the data in a specific way. Because it is a linear combination of the observed values Y_n sometimes it is called a linear smoother.

In particular, this predictor interpolates the data and at the interpolation points it has 0 prediction error. This is because

$$\Gamma(Y_n)\Lambda(Y_n) = \begin{bmatrix} \Gamma(Y_n, Y_{x_1})^T\Lambda(Y_n) \\ \cdots \\ \Gamma(Y_n, Y_{x_n})^T\Lambda(Y_n) \end{bmatrix} = I_n.$$

So that, for any j between 1 and n, $\Gamma(Y_n, Y_{x_j})^T\Lambda(Y_n)$ is a vector with zeroes everywhere except for the j^{th} entry which is 1. Use this in the prediction formula to get $\widehat{Y}_{x_j} = y_{x_j}$ and 0 on the prediction error.

In conclusion, we built an interpolating predictor that is optimal if we assume a Gaussian prior. This predictor and its prediction error have explicit formulas and are relatively simple to interpret. On the other hand we did not give an idea of what is the meaning of this prior assumption and this is what we discuss in the next section.

2.1.1 Modeling with Gaussian processes

Choosing a Gaussian prior is a very strong statement and at first sight its meaning seems difficult to understand. The object of this section is to show how it is possible to express modeling assumptions in an intuitive way by choosing particular mean and covariance functions.

For this part, we consider 1-dimensional inputs so let $\mathcal{X} \subset \boldsymbol{R}$ and

$$Y_x \sim GP(\mu(x), \Gamma(x, x')).$$

The mean function

The most simple mean function we can consider is a constant μ. The effect of changing this constant is to shift the realizations or paths of Y_x. Figure 2.1 below shows several realization of two G.P.s with the same covariance and different constant means.

If we want to model a general trend for the process we can choose a different function. For example we could write $\mu(x) = \mu x$ for a linear trend with slope μ. More generally, a popular choice is to define $\mu(x)$ as a linear combination of functions of the inputs like $\mu^T f(x) = \mu_1 + \mu_2 x + \mu_2 x^3 + \mu_3 \sin(x)$.

Since we are interested in estimating the mean of the process and we will derive some complex looking formulas we consider a constant mean function to simplify the notation. Nonetheless, all the computations can

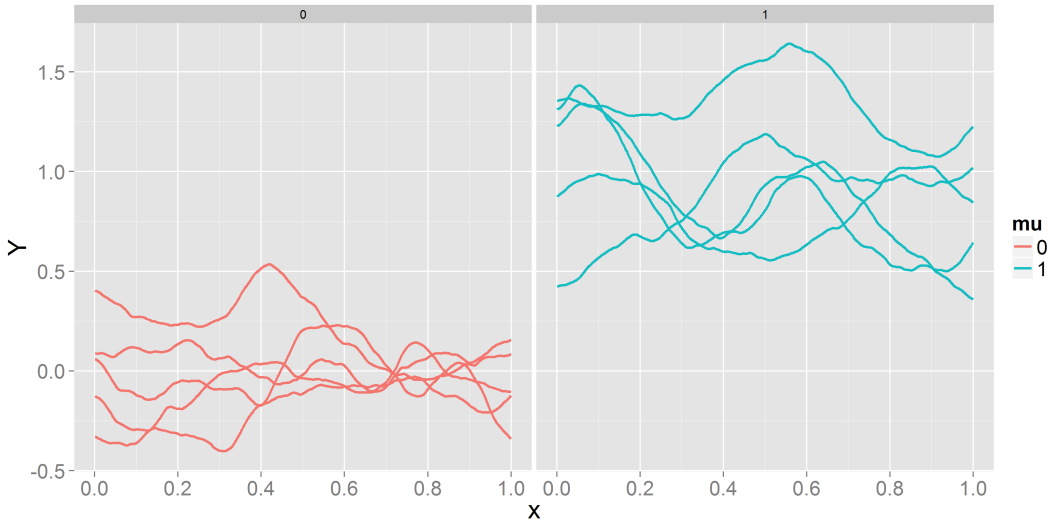


Figure 2.1: The outcomes of two Gaussian processes with the same covariance function and different constant means.

be generalized without much problem to the $\mu^T f(x)$ case. Also the constant mean has a very definite interpretation that a complex linear combination might lack.

The covariance function

The covariance function is the most important component of the process. It has to be a positive definite function, meaning that any covariance matrix built with it should be positive definite. Sometimes we call this function a kernel.

When we introduced Gaussian processes we talked about the fact that any finite subset of indexes should be coherent. A result due to A.Komogorov, called the Daniell-Kolmogorov extension theorem, says that this condition is enough for the process to exist. This theorem does not give any idea of the regularity of realizations of such a process. A second result by Kolmogorov, sometimes called the continuity theorem, says that if for any x, x'

$$E[|Y_x^{(k)} - Y_{x'}^{(k)}|^2] \leq \|x - x\|^{2(\alpha-k)}$$

where k is the order of derivation and α a parameter, then the paths of our Gaussian process are continuous and γ-Holder for any $0 < \gamma < \alpha - k$. Some covariance structures imply this condition. For example in [53] two popular choices, the Matern and Exponential, are studied. They are defined by

$$\Gamma_{M.3/2}(x, x') := \left(1 + \sqrt{3}|x - x'|\right) \exp\left(-\sqrt{3}|x - x'|\right)$$

and

$$\Gamma_{EXP}(x, x') = \exp\left(-|x - x'|\right).$$

Figure 2.2 shows the difference between the paths of processes with Matern and Exponential covariances.

Although the regularity of the process is a very interesting subject there are some other properties that can be encoded in the form of the covariance.

These other properties are defined by using an algebra of kernels. Recall that the only condition we required for a covariance function was positive definiteness. Some properties are

- **Scaling the outputs** the outputs can be scaled by a positive number σ^2. The resulting function, $\sigma^2\Gamma(x, x')$, is a kernel. Because $Var(Y_x) = \sigma^2\Gamma(x, x)$, we call σ^2 the variance parameter.
- **Scaling the inputs** The inputs can be scaled too. We get the kernel $\Gamma\left(\phi(x), \phi(x')\right)$ for any ϕ. If $\phi(x) = \dfrac{x}{\theta}$ we call θ the covariance parameter. Figure 2.2 shows the effect of this number on the paths. A bigger θ means less variation.

  Some nonlinear transformations are used too. For example $\phi(x) = x^2$ gives paths that vary first slowly and then faster. If we use $\phi(x) = \sin(x)$ we get periodic paths.
- **Addition** The sum of two kernels Γ_1 and Γ_2 is a kernel. If the kernel

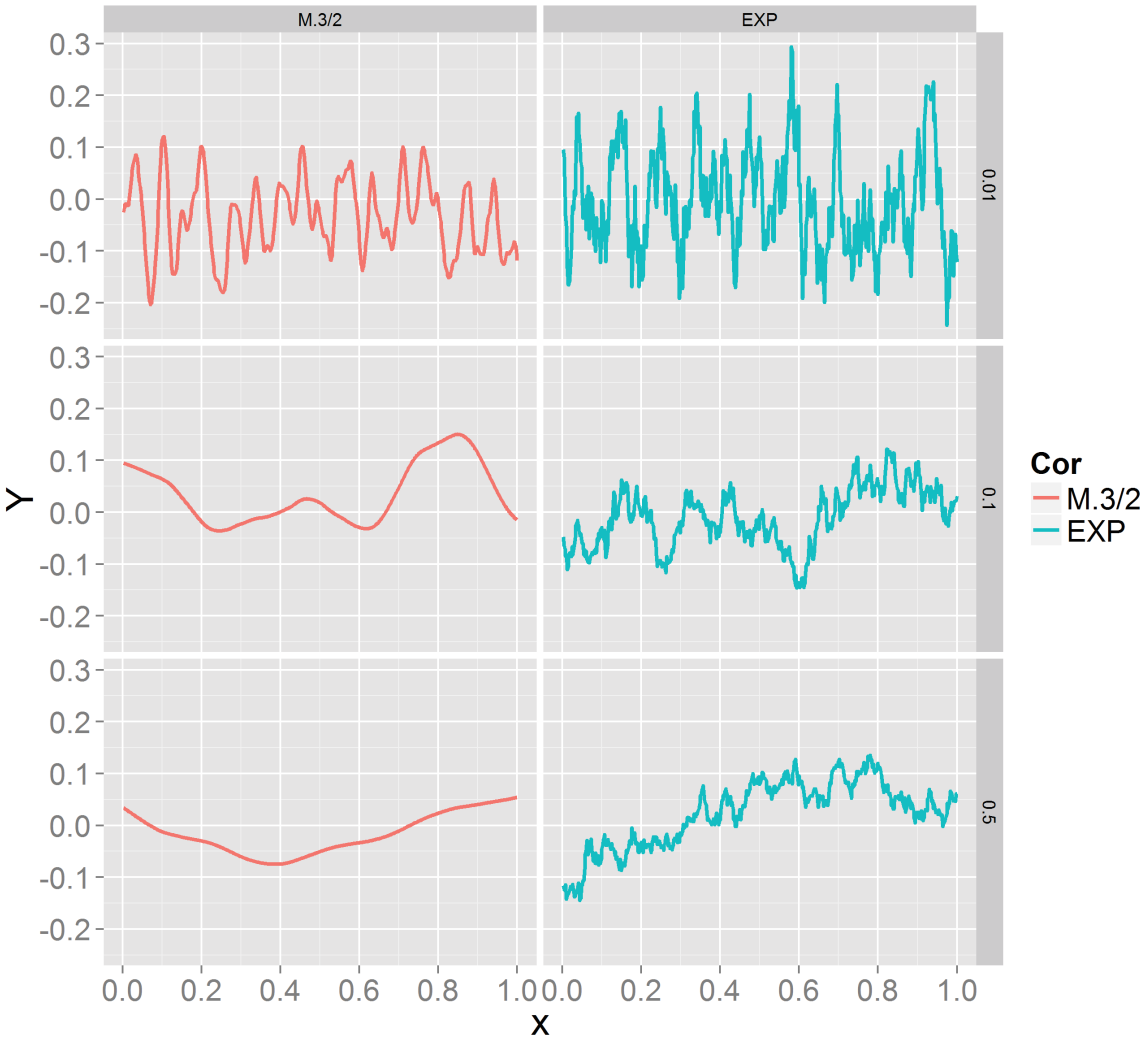


Figure 2.2: Each column represents a covariance: Matern 3/2 on the left and exponential on the right. The rows represent 3 covariance parameters 0.01, 0.1 and 0.5.

is thought as a measure of similarity, the sum kernel $\Gamma_1 + \Gamma_2$ measures it by either Γ_1 or Γ_2.

- **Features** $\Gamma(x, x') := \phi(x)^T \phi(x')$ is a kernel for any vector of features $\phi(x) := [\phi_1(x), ..., \phi_l(x)]$. For example $\phi(x) = [1, x, x^2]$. In fact it is possible to show that a function is kernel if and only if it can be expressed as an inner product of features. This features can encode a global form for the function.

  Another kernel we can build using a vector of features is $\Gamma(x, x') := \phi(x)^T \Gamma'(x, x') \phi(x')$ where Γ' is a kernel.

These are the properties we are going to use, but there are a lot more. For example [19] proposes an algorithm that uses this algebra to build covariances from descriptions made in English. For details on other properties you can consult [26], [47] or the webpage [25].

Our kernels usually have a variance and a covariance parameter. If we want to highlight this parametric form we use the slightly longer notation $Y_x \sim GP(\mu, \sigma^2 \gamma(x, x'; \theta))$.

Multidimensional inputs

If the process is indexed by vectors in $\mathbf{R}^D$ we have some options.

The first thing we can do is to build a kernel over the D-dimensional inputs but using a little more structure can be more easy to interpret and implement.

We can assume that γ is the product of 1-dimensional symmetric and positive definite functions

$$\gamma(x, x'; \theta) = \prod_{d=1}^{D} \gamma_d(x^d, x'^d; \theta^d).$$

Then, the correlation function depends on a vector of unknown parameters $\theta^T := [\theta^1, ..., \theta^D]^T$.

This is a very popular choice when the dimension of the inputs is small. Sometimes these models are said to have an automatic relevance determination. The size of the covariance parameters determine the relevance of each dimension. Big θ^ds imply small variations of the process along the input x^d. This concept was introduced by Neil Radford in his PHD thesis [45].

Another nice property of a GP with this kind of covariance is that if we use an exponential kernel in each dimension, then the paths are dense in the space of continuous functions. This is only true for particular choices of kernels like γ_{EXP}, see [39] for details. This does not mean that we can learn any continuous function efficiently, in fact the learning rate can be very slow as shown in [53].

Another important drawback of this kind of models is that if D is big so is the number of covariance parameters.

When the dimension of the inputs is large we can consider an additive model, like in [17], in which the covariance is

$$\gamma(x, x'; \theta) = \sum_{d=1}^{D} \gamma_d(x^d, x'^d; \theta^d).$$

This kind of models are useful mainly because of two reasons: they are computationally less expensive and they can be used to find functional relationships between the inputs.

A possible generalization for these additive models is to consider a functional ANOVA kernel

$$\gamma(x, x'; \theta) = \sum_{d=1}^{D} \gamma_d(x^d, x'^d; \theta^d) + \sum_{i=1}^{D} \sum_{j=1}^{D} \gamma_{ij}(x^{(i,j)}, x'^{(i,j)}; \theta^{(i,j)}) + \cdots$$

where we add higher order interaction kernels.

Which kernel?

Using a Gaussian prior looks like a restrictive hypothesis but we can model many different types of functions in very interesting ways by building a kernel. In practice deciding what is the appropriate one is still an open question. Normally, we choose the kernel in advance, this is we choose a class of models, that depend on a small number of parameters that we need to fix.

In particular we focus on kernels with variance and covariance parameters. In the next section we describe three methods to fix μ, σ^2 and θ.

2.2 Model fitting via parameter estimation

Pick a kernel γ and consider the family of Gaussian processes

$$Y_x \sim GP(\mu, \sigma^2\gamma(x, x'; \theta))$$

indexed by μ, σ^2 and θ. We observe the process at some input points $(x_1, Y_{x_1}), ..., (x_n, Y_{x_n})$ and we would like like to estimate a set of parameters using this data to build the optimal linear predictor of the last section.

Here we are going to describe three ways to do this. the maximum likelihood principle, that selects the parameters that make the data more likely; a Bayesian model selection, that assumes a prior distribution on each of the parameters to get a posterior distribution and a cross-validation technique, that minimizes the training error over several partitions of the data set.

As before we note Y_n the vector of observed values. Using an analogous notation to the one in previous sections we note $\Gamma(n; \theta) = \sigma^2\gamma(n; \theta)$ the

covariance matrix of the observations and $\Lambda(n;\theta) = \frac{1}{\sigma^2}\lambda(n;\theta)$ its inverse.

Sometimes, it is useful to put all the parameters into a single vector that we note ξ.

We use capital letters to represent random variables and vectors of random variables. Because we work with several distributions that we need to evaluate at different points we write $f_{Y_x}(s;\xi)$ to say that we evaluate the probability density function or pdf of the random variable Y_x defined by the parameters ξ at s.

2.2.1 Maximum likelihood estimation

The likelihood function for a given vector of observations $Y_n = y_n$ is defined as

$$L_{Y_n}(\mu,\sigma^2,\theta;y_n) = \frac{1}{|2\pi(\sigma^2)^n\gamma(n;\theta)|^{1/2}} \exp\left(-\frac{1}{2\sigma^2}(y_n-\mu\mathbf{1})^T\lambda(n;\theta)(y_n-\mu\mathbf{1})\right).$$

where $\mathbf{1}$ is a vector of n ones. This equation is a function of the parameters that depends on the data.

The goal of this method is to use the parameters that maximize the likelihood of the data as an estimate of the true parameters. Then, the maximum likelihood estimators are the parameters that make the event $\{Y_n = y_n\}$ more likely.

In our case, we work with the logarithm of the likelihood function. It is

$$\log L_{Y_n}(\xi;y_n) = -\frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(y_n-\mu\mathbf{1})^T\lambda(n;\theta)(y_n-\mu\mathbf{1}) - \frac{1}{2}\log|2\pi\gamma(n;\theta)|.$$

For a given θ, we define the maximum likelihood estimator for the mean and variance parameters as the unique values for which their corresponding

partial derivatives are zero. The estimators are

$$\widehat{\mu}_{MLE}(\theta) = \left(\mathbf{1}^T\lambda(n;\theta)\mathbf{1}\right)^{-1}\left(\mathbf{1}^T\lambda(n;\theta)y_n\right)$$

for the mean and

$$\widehat{\sigma}^2_{MLE}(\theta) = \frac{(y_n - \widehat{\mu}_{MLE}(\theta)\mathbf{1})^T\lambda(n;\theta)(y_n - \widehat{\mu}_{MLE}(\theta)\mathbf{1})}{n}$$

for the variance parameter.

The estimator for the covariance parameter is defined as the value that maximizes the logarithm of the likelihood after pluggin-in the other two estimators. The resulting function depends only on θ and it is defined as

$$\log L_{Y_n}(\widehat{\mu}_{MLE}(\theta), \widehat{\sigma}^2_{MLE}(\theta), \theta, y_n) = -\frac{n}{2}\log\widehat{\sigma}_{MLE}(\theta)^2 - \frac{1}{2}\log|\gamma(n;\theta)|.$$

To compute these estimates we start with the covariance parameter estimator $\widehat{\theta}_{MLE}$. Then, we use this estimate to compute $\widehat{\mu}_{MLE}(\widehat{\theta}_{MLE})$ and finally $\widehat{\sigma}^2_{MLE}(\widehat{\theta}_{MLE})$.

The plug-in prediction is obtained by computing the optimal linear prediction using the estimates defined above. The prediction at a point x when $\{Y_n = y_n\}$ is

$$\widehat{Y}_x = \widehat{\mu}_{MLE}(\widehat{\theta}_{MLE}) + \gamma\left(n, x; \widehat{\theta}_{MLE}\right)^T\lambda\left(n; \widehat{\theta}_{MLE}\right)(y_n - \widehat{\mu}_{MLE}(\widehat{\theta}_{MLE})\mathbf{1})$$

with prediction error

$$E[|Y_x - \widehat{Y}_x|^2] = \widehat{\sigma}^2_{MLE}(\widehat{\theta}_{MLE}) \left(1 - \gamma\left(n, x; \widehat{\theta}_{MLE}\right)^T\lambda\left(n; \widehat{\theta}_{MLE}\right)\gamma\left(Y_n, Y_x; \widehat{\theta}_{MLE}\right)\right).$$

Sometimes in practice an additional term is added to the prediction error. This term comes from the predictive distribution obtained by setting particular priors on the parameters. The idea is that this new variance takes into account a particular form of uncertainty we have on the variance and mean parameters. We describe this method in the next section.

2.2.2 Bayesian model selection

For this kind of model selection or inference, we assume that the parameters are random variables. We fix a prior distribution that reflect our beliefs before observing the data. Then, we update this prior using the data. Computing this posterior distribution is the goal of this method.

We note M, Σ^2 and Θ the random variables representing the mean, variance and the vector of covariance parameters. We assume that these random variables are continuous. Using Bayes' rule we have that the posterior distribution is

$$f_{M,\Sigma^2,\Theta|Y_n}(\mu, \sigma^2, \theta|y_n) = \frac{f_{Y_n|M,\Sigma^2,\Theta}(y_n|\mu, \sigma^2, \theta) f_{M,\Sigma^2,\Theta}(\mu, \sigma^2, \theta)}{f_{Y_n}(y_n)} \tag{2.3}$$

The distribution on the denominator is the normalizing constant needed to have a pdf. It is called the marginal likelihood. We can compute it by solving

$$f_{Y_n}(y_n) = \int f_{Y_n|M,\Sigma^2,\Theta}(y_n|\mu, \sigma^2, \theta) f_{M,\Sigma^2,\Theta}(\mu, \sigma^2, \theta) \, \mathrm{d}\mu \, \mathrm{d}\sigma^2 \, \mathrm{d}\theta.$$

So, to deduce the posterior distribution we need to define the distribution of $[Y_n|M, \Sigma^2, \Theta]$ and the joint distribution of (M, Σ^2, Θ).

We assume that

$$[Y_x|M = \mu, \Sigma^2 = \sigma^2, \Theta = \theta] \sim GP(\mu, \sigma^2\gamma(x, x'; \theta))$$

so $[Y_n|M, \Sigma^2, \Theta]$ is a Gaussian vector. For the joint distribution we have two options. Either we set a full joint prior or we give a marginal and some conditional distributions - for example $f_{M|\Sigma^2,\Theta}(\mu|\sigma^2, \theta) f_{\Sigma^2|\Theta}(\sigma^2|\theta) f_{\Theta}(\theta)$.

If we can compute the posterior distribution we can set the parameters as the values that maximize this distribution. This is called a maximum a posteriori or MAP estimation. But instead of using only a point estimate we can use the distribution to compute an average over the parameters.

We define the predictive distribution as

$$f_{Y_x|Y_n}(y_x|y_n) = \int f_{Y_x|Y_n,M,\Sigma^2,\Theta}(y_x|y_n,\mu,\sigma^2,\theta) f_{M,\Sigma^2,\Theta}(\mu,\sigma^2,\theta)\ \mathrm{d}\mu\ \mathrm{d}\sigma^2\ \mathrm{d}\theta.$$

This distribution can be used to make a prediction at x and to compute the variance of this prediction.

In principle we can choose any prior and then compute the marginal likelihood, the posterior and predictive distributions using several methods like Expectation Propagation, Markov Chain Monte Carlo sampling, or variational Bayesian ideas. We do not discuss these here although they are very prominent techniques in Bayesian inference.

In practice usually the covariance parameter is supposed to be known or its MLE estimator is used and the inference is done on M only or on (M, Σ^2). In these cases, two types of priors are used, non informative and conjugate. On appendix A we summarize some popular choices.

The posteriors related to the non informative priors involve the maximum likelihood estimators, see appendix A. The posterior distributions for the conjugate priors have the same form as the priors with new parameters - this is the definition of conjugate prior. They can be found in A along with their corresponding predictive distributions.

2.2.3 Cross validation

As in the methods above, we are given a vector of observations Y_n drawn from a Gaussian process $\{Y_x : x \in \mathcal{X}\}$. Here we assume that the mean parameter is equal to 0, there is no cross validation estimator for the mean parameter. Then the process is defined by the usual set of parameters $\xi = [0, \sigma^2, \theta^1, ..., \theta^D]^T$.

The idea of cross validation is to use different partitions of the set of observations. A first approach is to split the observations into two separate

sets $Y_{-1} := [Y_{x_1}, ..., Y_{x_l}]^T$ and $Y_1 := [Y_{x_{l+1}}, ..., Y_{x_n}]^T$, a training set and a validation set. We use the training set to build the optimal linear predictor formula of the validation set as a function of the unknown parameters ξ. We note this formula $\widehat{Y}_{-1}$. We select the parameters as the argument that minimizes the prediction error over the validation set $E[|\widehat{Y}_{-1} - Y_1|^2]$.

If we use only one partition we risk having an estimate that depends too much on the form of the partition. Cross validation generalizes the idea of the validation procedure by considering many 2-way partitions of the observations and defining the risk of choosing a particular ξ as the average risk over all the partitions.

The way we formalize this idea is by first dividing the set of observations into V disjoint parts $Y_1, ..., Y_V$. Then, for all i between 1 and V, we set $Y_{-i} := (Y_1 \cup ... \cup Y_V) \setminus Y_i$ as the training set and Y_i as the validation set. The average risk is then written as $R(\xi, Y_n; V) := \frac{1}{V} \sum_{i=1}^{V} E[|\widehat{Y}_{-i} - Y_i|^2]$.

Deciding how many partitions V is sometimes difficult, to avoid this problem usually V is set to n so that the training set contains all the observations except for one. This is the method we consider. It is usually called a leave one out cross-validation.

When the vector of observations is equal to some real values $y_n := [y_{x_1}, ..., y_{x_n}]^T$, then $E[|\widehat{Y}_{-i} - Y_i|^2]$ becomes $|\widehat{y}_{-i} - y_i|^2$ and the average risk does not depend on the variance parameter σ^2. Then, our estimates for θ are defined as

$$\widehat{\theta} := \arg\max_{\theta} R(\xi, y_n; n) = \arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} |\widehat{y}_{-i} - y_i|^2.$$

For the variance parameter we do something a little more complex. Minimizing the average risk makes sense if it is a good approximation of the true risk uniformly in ξ. The way we estimate σ^2 is by considering the average of the ratio between the validation risk and the true risk associated to each of the partitions. The true risk is approximated by the prediction

error at Y_i of the optimal linear predictor built using Y_{-i} and $\widehat{\theta}$. Then, the ratio for observation Y_i being y_i is

$$\frac{|\widehat{y}_{-i} - y_i|^2}{E[|\widehat{Y}_{-i} - Y_i|^2]} = \frac{|\widehat{y}_{-i} - y_i|^2}{\sigma^2 \left(1 + \gamma(-i, i; \widehat{\theta})^T \lambda(-i; \widehat{\theta}) \gamma(-i, i; \widehat{\theta})\right)}.$$

The argument used in [3] is the following: for the true variance parameter this ratio should be close to 1. Then, the variance parameter selected is

$$\widehat{\sigma^2} := \frac{1}{n} \sum_{i=1}^{n} \frac{|\widehat{y}_{-i} - y_i|^2}{1 + \gamma(-i, i; \widehat{\theta})^T \lambda(-i; \widehat{\theta}) \gamma(-i, i; \widehat{\theta})}.$$

At first sight this procedure seems slower than maximizing the likelihood since for every observation we have to compute the inverse of $\gamma(-i; \theta)$ but in fact by using a blockwise inversion formula for $\gamma(n; \theta)$ it is possible to write the inverse of the partial matrices as a function of the complete one. For more details one can consult the PHD of F. Bachoc [3].

For a clear exposition of cross-validation in general one can consult one of the many articles by S. Arlot, for example [1].

Now, to make a prediction at an unobserved point x, we plug-in the estimated parameters in a prediction built using the whole set of observations Y_n. The prediction error is the plug-in prediction error of the optimal linear predictor built with the whole set of observations.

Which method of estimation?

We presented 3 popular estimation methods. The first method produces point estimates with explicit formulas without making additional assumption on the model. It only uses the data.

The second method adds a lot of structure but at the end we get a distribution for the parameters. We can use this distribution to analyze the data and compute a predictive distribution that is different from that

of $[Y_x|Y_n]$ in the other two cases.

The last method produces point estimates. This time they are based on minimizing the mean squared error.

A common drawback of any of these three methods is the complexity of the computations. When we work with Gaussian processes we need to invert the covariance matrix of the observations and this is done in $O(n^3)$ operations. This means that these models are not very well suited for high dimensional problems. There are some approximate inference procedures, see [44], that address this problems in which the prior distribution is modified. Another possibility is to use a variational method.

Also, the properties of the parameter estimators are not fully understood as is described in [3], [29] and [32]. The main concern is the consistency of the estimators and the predictive performance. Although most of the results are asymptotic it is important to remember that finite sample properties can be very different, see [53] for a study of the learning rate and [29] for simulations regarding the estimation of the covariance parameter.

2.2.4 Gaussian process models

When we assume a Gaussian process prior we end up with a distribution that models functions that pass through the observed points. We can use this distribution to do things that we would do with the unknown function from which we sampled the data.

For example, we can use this distribution to decide where we should add new observations by looking at the variance of the posterior distribution. This idea has been used to device criteria to explore the function to find a global minimum or to determine for which inputs the observed data can be higher than a threshold. See for example [49] and their references.

Another possible use for this distribution is to determine the effect that

varying an input has on the output. This can be used as a method to classify and select the variables that influence the model the most to study them or to reduce the dimension of the inputs, see for example [28].

2.3 Multi-fidelity regression with Gaussian processes

Multi-fidelity is a concept related to applications in which the data we observe is very hard to produce. This data can be the result of an experiment or of a computer simulation.

Most of the time in this cases a more simple experiment is available to, for example, have performance estimates or to check individual parts of the model. The simplifications can be very different in nature, we can think of omitted parts/physics; coarser meshes; reduced basis models; or old versions of the code that has been extensively studied and for which we dispose of a lot of observations.

At any rate there are two types of models: the complicated one, the one we want to understand and all the available simplifications. This means that we dispose of two types of data: high and lower fidelity.

Multi-fidelity is about using data obtained from some of the simplified models and use it along the data that is hard to produce to enhance the estimation, inference, or predictions related to high-fidelity model.

2.3.1 Gaussian processes for multi-fidelity

We assume that all the data have the same input space and the same output or target quantity. With this in mind we use an additional index to distinguish between each data set and write $Y_{j,n_j} = \{Y_{j,x} : x \in \mathcal{X}_j\}$ to denote the observations over the set $\mathcal{X}_j$ that contains n_j indexes obtained from the experiment j.

Before moving on we need to introduce some notation. All the lower fidelity data is be modeled by Gaussian processes. For each j we assume the prior

$$Y_{j,x} \sim GP(\mu_j, \sigma_j^2 \gamma_j(x, x'; \theta_j))$$

with a constant mean parameter and a variance and covariance parameters. As before, when needed, we group all the parameters into a single vector ξ_j.

We use one last bit of notation regarding the vector of observations. We write $Y_j(n_k)$ to denote the observations of Y_j made over the index set of Y_{k,n_k}.

Additive models

The first model in the context of Gaussian processes was proposed in [30] by Marc Kennedy and Tony O'Hagan. Suppose we have N data sets, then they assumed three things. The first is that the models can be ordered according to their fidelity, this is we are given an order for the processes. In this sense we use an ascending order on the indexes so that the biggest index is for the target process.

The second assumption is that the lowest fidelity process is

$$Y_{1,x} \sim GP(\mu_1, \sigma_1 \gamma_1(x, x')).$$

The third assumption, that is in fact $N - 1$ assumptions, is that there is another Gaussian process

$$Y_{d_1,x} \sim GP(\mu_{d_1}, \sigma_{d_1} \gamma_{d_1}(x, x')).$$

that is independent of Y_1 and a function $g_1(x)$ such that

$$Y_{2,x} = g_1(x) Y_{1,x} + Y_{d_1,x}.$$

Because we are working with Gaussian processes, we have that

$$Y_{2,x} \sim GP(g_1(x)\mu_1 + \mu_{d_1},\ g_1(x)g_1(x')\sigma_1^2\gamma_1(x,x') + \sigma_{d_1}^2\gamma_{d_1}(x,x')). \quad (2.4)$$

Now for $Y_{2,x}$ there is a Gaussian process $Y_{d_2,x}$ independent of $Y_{2,x}$ and a function $g_2(x)$ such that

$$Y_{3,x} = g_2(x)Y_{2,x} + Y_{d_2,x}.$$

Like so, we define the next level until we get to the N^{th} and final one which is a Gaussian process too with the form of equation (2.4).

Properties of the model

First we have that

$$\begin{aligned} Cov(Y_{j+1,x}, Y_{j,x'}|Y_{j,x}) &= g(x)Cov(Y_{j,x}, Y_{j,x'}|Y_{j,x}) + Cov(Y_{d_j,x}, Y_{j,x'}|Y_{j,x}) \\ &= Cov(Y_{d_j,x}, Y_{j,x'}|Y_{j,x}) \end{aligned}$$

and $Cov(Y_{d_j,x}, Y_{j,x'}|Y_{j,x}) = 0$ because

$$\begin{aligned} f_{Y_{d_j,x},Y_{j,x'}|Y_{j,x}}(u,v|w) &= \frac{f_{Y_{d_j,x},Y_{j,x'},Y_{j,x}}(u,v,w)}{f_{Y_{j,x}}(w)} \\ &= \frac{f_{Y_{d_j,x}}(u)f_{Y_{j,x'},Y_{j,x}}(v,w)}{f_{Y_{j,x}}(w)} \\ &= f_{Y_{d_j,x}}(u)f_{Y_{j,x'}|Y_{j,x}}(v|w). \end{aligned}$$

So as long as $Y_{d_j,x}$ is independent of $Y_{j,x'}$ for any x, x', which is what we meant by independent processes before, then we have that

$$Cov(Y_{j+1,x}, Y_{j,x'}|Y_{j,x}) = 0.$$

The other implication is true for Gaussian processes, namely if the conditional expectation is zero, we can write

$$\begin{aligned} Cov(Y_{j+1,x}, Y_{j,x'}|Y_{j,x}) &= Cov(Y_{j+1,x}, Y_{j,x'}) \\ &\quad - Cov(Y_{j+1,x}, Y_{j,x})Cov(Y_{j,x}, Y_{j,x})^{-1}Cov(Y_{j,x}, Y_{j,x'}) \\ &= 0 \end{aligned}$$

and set

$$\begin{aligned} g_j(x) &:= Cov(Y_{j+1,x}, Y_{j,x})Cov(Y_{j,x}, Y_{j,x})^{-1} \\ Y_{d_j,x} &:= Y_{j+1,x} - g_1(x)Y_{j,x}. \end{aligned}$$

so that $Y_{d_j,x}$ is independent of $Y_{j,x}$ by construction.

Kennedy and O'Hagan interpret the fact that the conditional covariance is 0 as if $Y_{j,x}$ contains all the information about $Y_{j+1,x}$ we can get from Y_j. If you want this condition to hold for Gaussian processes this is equivalent to assume the model we described above.

In summary, the model we assume for N data sets is built from the bottom up and is such that for any j there is a Gaussian process Y_{d_j} independent of Y_j and a function $g_j(x)$ that verify

$$Y_{j+1,x} = g_j(x)Y_{j,x} + Y_{d_j,x}.$$

For simplicity from now on we work with $N = 2$. The next property of the model will be useful to build the regression function with its prediction error and to perform model selection both sequentially from the bottom up.

Proposition 1. *If we observe $Y_1(n_2)$ in Y_{1,n_1} and we note $f_{Y_{n_1},Y_{n_2}}(y_{n_1}, y_{n_2})$ the joint density function of the observations at (y_{n_1}, y_{n_2}). Then,*

$$f_{Y_{1,n_1},Y_{2,n_2}}(y_{n_1}, y_{n_2}) = f_{Y_{1,n_1}}(y_{n_1})f_{Y_{d_1}(n_2)}(y_d(n_2))$$

where $y_{d_1}(n_2) := y_{n_2} - g_1(n_2)y_1(n_2)$.

Proof. We make the change of variables $[Y_{1,n_1}, Y_{2,n_2}] \to [Y_{1,n_1}, Y_{d_1}(n_2)]$ and write

$$\begin{aligned} f_{Y_{1,n_1},Y_{2,n_2}}(y_{n_1}, y_{n_2}) &= f_{Y_{1,n_1},Y_{d_1}(n_2)}(y_{n_1}, y_{d_1}(n_2)) \left| \frac{\partial(y_{n_1}, y_{d_1}(n_2))}{\partial(y_{n_1}, y_{n_2})} \right| \\ &= f_{Y_{1,n_1}}(y_{n_1}) f_{Y_{d_1}(n_2)}(y_{d_1}(n_2)). \end{aligned}$$

because the absolute value of the determinant of the Jacobian of the change of variables, that we note $\left| \frac{\partial(y_{n_1}, y_{d_1}(n_2))}{\partial(y_{n_1}, y_{n_2})} \right|$, is 1. □

This is a very useful property. The first remark is that the property does not depend on the fact that we are working with Gaussian processes.

A second remark is that we need to have a nested set of observations for this property to hold, i.e. $\mathcal{X}_2 \subset \mathcal{X}_1$. Otherwise we would have $y_{d_1}(n_2) := y_{n_2} - g_1(n_2) y_1(n_2)$ defined in terms of the unobserved vector $y_1(n_2)$ and the joint distribution would be a convolution which is difficult to work with.

In particular this property implies that the likelihood of an appropriate data set is decomposed when written in terms of Y_1 and Y_{d_1}. This means that if we want to infer the parameters of Y_2, we can infer those of Y_1 and Y_{d_1} independently and then put them together according to the formulas in (2.4). This is coherent with the way the model was built: from the bottom up.

We can use this proposition to write the mean and variance of $Y_{2,x}|Y_{n_1}, Y_{n_2}$ in terms of Y_1 and Y_{d_1}.

Proposition 2. *If we observe $Y_1(n_2)$ in Y_{1,n_1} and if we note*

$$\begin{aligned} Y_{1,x}|Y_{1,n_1} &\sim GP\left(\widehat{Y}_{1,x}, \widehat{\Gamma}_1(x, x')\right) \\ Y_{d_1,x}|Y_{d_1}(n_2) &\sim GP\left(\widehat{Y}_{d_1,x}, \widehat{\Gamma}_{d_1}(x, x')\right) \end{aligned}$$

Then,

$$Y_{2,x}|Y_{1,n_1}, Y_{2,n_2} \sim GP\left(g_1(x)\widehat{Y}_{1,x} + \widehat{Y}_{d,x}, g_1(x)\widehat{\Gamma}_1(x,x')g_1(x') + \widehat{\Gamma}_{d_1}(x,x')\right).$$

Proof. We begin with the conditional expectation and write

$$E[Y_{2,x}|Y_{1,n_1}, Y_{2,n_2}] = g_1(x)E[Y_{1,x}|Y_{1,n_1}, Y_{2,n_2}] + E[Y_{d_1,x}|Y_{1,n_1}, Y_{2,n_2}].$$

We focus on the first term and rewrite it using an integral form

$$\begin{aligned}
E[Y_{1,x}|Y_{1,n_1}, Y_{2,n_2}] &= \int s\ f_{Y_{1,x}|Y_{1,n_1},Y_{2,n_2}}(s|Y_{1,n_1}, Y_{2,n_2})\ \mathrm{d}s \\
&= \int s\ \frac{f_{Y_{1,x},Y_{1,n_1},Y_{2,n_2}}(s, Y_{1,n_1}, Y_{2,n_2})}{f_{Y_{1,n_1},Y_{2,n_2}}(Y_{1,n_1}, Y_{2,n_2})}\ \mathrm{d}s \\
&= \int s\ \frac{f_{Y_{1,x},Y_{1,n_1}}(s, Y_{1,n_1})}{f_{Y_{1,n_1}}(Y_{1,n_1})}\ \frac{f_{Y_{d_1}(n_2)}(Y_{2,n_2} - g_1(n_2)Y_1(n_2))}{f_{Y_{d_1}(n_2)}(Y_{2,n_2} - g_1(n_2)Y_1(n_2))}\ \mathrm{d}s.
\end{aligned}$$

Since $f_{Y_{d_1}(n_2)}(Y_{2,n_2} - g_1(n_2)Y_1(n_2)) \neq 0$, we have that

$$E[Y_{1,x}|Y_{1,n_1}, Y_{2,n_2}] = E[Y_{1,x}|Y_{1,n_1}].$$

We can do the same for the other part of the sum, $E[Y_{d,x}|Y_{1,n_1}, Y_{2,n_2}]$, and write the expectation at the beginning as

$$\begin{aligned}
E[Y_{2,x}|Y_{1,n_1}, Y_{2,n_2}] &= g_1(x)E[Y_{1,x}|Y_{1,n_1}] + E[Y_{d_1,x}|Y_{d_1}(n_2)] \\
&= g_1(x)\widehat{Y}_{1,x} + \widehat{Y}_{d_1,x}.
\end{aligned}$$

For the covariance we write

$$\begin{aligned}
Cov(Y_{2,x}, Y_{2,x'}|Y_{1,n_1}, Y_{2,n_2}) &= Cov(g_1(x)Y_{1,x} + Y_{d,x}, g_1(x')Y_{1,x'} + Y_{d_1,x'}|Y_{1,n_1}, Y_{2,n_2}) \\
&= g_1(x)g_1(x')Cov(Y_{1,x}, Y_{1,x'}|Y_{1,n_1}, Y_{2,n_2}) \\
&\quad + Cov(Y_{d_1,x}, Y_{d_1,x'}|Y_{1,n_1}, Y_{2,n_2}).
\end{aligned}$$

We take a look at the first term $g_1(x)g_1(x')Cov(Y_{1,x}, Y_{1,x'}|Y_{1,n_1}, Y_{2,n_2})$. It is equal to

$$g_1(x)g_1(x')E[(Y_{1,x} - E[Y_{1,x}|Y_{1,n_1}, Y_{2,n_2}])(Y_{1,x'} - E[Y_{1,x'}|Y_{1,n_1}, Y_{2,n_2}])|Y_{1,n_1}, Y_{2,n_2}].$$

We can rewrite the conditional expectations on the inside as in the first part of the proof to get

$$g_1(x)g_1(x')E[(Y_{1,x} - E[Y_{1,x}|Y_{1,n_1}])(Y_{1,x'} - E[Y_{1,x'}|Y_{1,n_1}])|Y_{1,n_1}].$$

Which is equal to $g_1(x)\widehat{\Gamma}_1(x,x')g_1(x')$. The second part of the covariance can be analyzed in the same way. Finally

$$Cov(Y_{2,x}, Y_{2,x'}|Y_{1,n_1}, Y_{2,n_2}) = g_1(x)\widehat{\Gamma}_1(x,x')g_1(x') + \widehat{\Gamma}_{d_1}(x,x').$$

□

We showed that to make a prediction on $Y_{2,x}$ using this model, we make two independent predictions first, one for $Y_{1,x}$ and the second for $Y_{d_1,x}$, and the proposition above tells us how we should put those together. Also the prediction error can be built by adding the prediction errors of the constituent processes.

Finally we can use this proposition to build a model where we observe N vectors drawn from N Gaussian processes such that

$$Y_{j+1,x} = g_j(x)Y_{j,x} + Y_{d_j,x}.$$

Where g_j is known and for each j in $\{1, ..., N-1\}$ and all x, x', $Y_{d_j,x}$ is independent of $Y_{j,x'}$.

In the next corollary, we use $[u_1, ..., u_n]^T \odot [w_1, ..., w_n]^T := [u_1w_1, ..., u_nw_n]^T$.

Corollary 1. *If we observe $Y_j(n_{j+1})$ in Y_{n_j} for every j between 1 and $N-1$ and note $y_{d_j}(n_{j+1}) := y_{n_{j+1}} - g_j(n_{j+1}) \odot y_j(n_{j+1})$. Then,*

$$f_{Y_{N,n_N}, Y_{N-1,n_{N-1}}, ..., Y_{1,n_1}}(y_{n_N}, y_{n_{N-1}}, ..., y_{n_1}) = f_{Y_{n_1}}(y_{n_1}) \prod_{j=1}^{N-1} f_{Y_{d_j}(n_{j+1})}(y_{d_j}(n_{j+1})).$$

Also, if we note

$$Y_{1,x}|Y_{1,n_1} \sim GP\left(\widehat{Y}_{1,x}, \widehat{\Gamma}_1(x,x')\right)$$

and

$$Y_{d_j,x}|Y_{d_j}(n_{j+1}) \sim GP\left(\widehat{Y}_{d_j,x}, \widehat{\Gamma}_{d_j}(x,x')\right).$$

Then, $Y_{k,x}$ *conditionally on* $[Y_{1,n_1},...,Y_{N,n_N}]$ *is a Gaussian process with mean*

$$G_1(x)\widehat{Y}_{1,x} + \sum_{j=1}^{N-1} G_{j+1}(x)\widehat{Y}_{d_j,x}$$

and covariance

$$G_1(x)\widehat{\Gamma}_1(x,x')G_1(x') + \sum_{j=1}^{N-1} G_{j+1}(x)\widehat{\Gamma}_{d_j}(x,x')G_{j+1}(x').$$

Where we define $G_l(x)$ *as the product* $\prod_{j=l}^{N-1} g_j(x)$ *and* $G_N(x) = 1$ *for any* x*.*

Proof. We can prove the first part by induction. For $N = 2$ the result is true. We assume that that for k we have

$$f_{Y_{k,n_k},Y_{k-1,n_{k-1}},...,Y_{1,n_1}}(y_{n_k}, y_{n_{k-1}}, ..., y_{n_1}) = f_{Y_{1,n_1}}(y_{n_1}) \prod_{j=1}^{k-1} f_{Y_{d_j}(n_{j+1})}(y_{d_j}(n_{j+1})).$$

For $k+1$ we make the following change of variables

$$[y_{n_1}, y_{n_2}, ..., y_{n_{k+1}}] \to [y_1(n_1), y_1(n_2), ..., y_{d_1}(n_{N-1})].$$

The Jacobian is diagonal with determinant equal to $g_1(x)^{k-1}/(G_2(x)\cdots G_{k-1}(x))$. Now we separate the part corresponding to $Y_{d_1}(n_k)$ to have

$$f_{Y_{1,n_1},Y_1(n_2),...,Y_1(n_k)}(y_{n_1}, y_1(n_2), ..., y_1(n_k))g_1(x)^{k-2}/(G_2(x)G_3(x)\cdots G_{k-2}(x))$$
$$f_{Y_{d_1}}(n_{k+1})(y_{d_k}(n_{k+1}))(g_1(x)/G_{k-1}(x)).$$

We change the variables on the first pdf back to the original ones; we obtain a determinant that cancels with the one of the first change of variables; use the recursion hypothesis and we change back the last term to obtain

the result.

For the second part notice that $E[Y_{k,x}|Y_{1,n_1}, ..., Y_{N,n_N}]$ can be written as

$$G_1(x)E[Y_{1,x}|Y_{1,n_1}, ..., Y_{N,n_N}] + \sum_{j=1}^{N-1} G_{j+1}(x)E[Y_{d_j,x}|Y_{1,n_1}, ..., Y_{N,n_N}].$$

We can rewrite each of the terms so that we get rid of some conditional random vectors to get the formula for the mean. For the covariance we can do an analogue proof. □

The idea behind these properties is to decompose the inference of the parameters and the computation of $E[Y_{N,x}|Y_{1,n_1}, ..., Y_{N,n_N}]$ into a sequence of steps. This reduces the number of operations needed to compute the formulas and gives us a series of intermediate predictions that might be interesting on themselves.

2.3.2 Building a multi-fidelity model based on G.P.s

The propositions above suggest a sequential strategy to infer the parameters of the model and to build the prediction and its error. This was remarked in [33] by Loic Le Gratiet.

The model we considered for N data sets is

$$Y_{j+1,x} = g_j(x)Y_{j,x} + Y_{d_j,x}$$

where Y_1 is a G.P. and each of the $Y_{d_j,x}$ is a G.P. independent of $Y_{j,x'}$ for all x, x'.

In this part we make the assumption that each of the functions g_j can be written as $\beta_j^T h_j(x)$ where β_j is a vector of unknown parameters and $h_j(x)$ is a vector of known functions.

Our objective is to describe a sequential strategy to infer the parameters

of Y_1 and those of all the difference processes Y_{d_j} as well as the scaling functions $g_j = \beta_j^T h_j(x)$. Once we have selected a model we can compute the regression function and its prediction error using corollary 1.

If the observed input sets are nested $\mathcal{X}_N \subset \cdots \subset \mathcal{X}_2 \subset \mathcal{X}_1$, the density function of $[Y_{n_1}, ..., Y_{n_k}]^T$ at $[y_{1,n_1}, ..., y_{N,n_N}]^T$ can be written as

$$f_{Y_{1,n_1}}(y_{n_1}) \prod_{j=1}^{N-1} f_{Y_{d_j}(n_{j+1})}(y_{d_j}(n_{j+1}))$$

where $y_{d_j}(j+1, n_{j+1}) := y_{n_{j+1}} - g_j(n_{j+1}) \odot y_j(n_{j+1})$. This functions is a multivariate Gaussian distribution. Choosing the variance and covariance parameters is essentially the same as when we were working with a single data set. The most important difference is on the mean parameters of the Y_{d_j}'s. To see why we follow the new vector of parameters β_j.

The vector β_j is only involved in the expression of $f_{Y_{d_j}(n_{j+1})}(y_{d_j}(n_{j+1}))$. It appears on the exponent, specifically in

$$\big(y_{d_j}(n_{j+1}) - \mu_{d_j}(n_{j+1})\big)$$

If we note $\mathbf{1}(n_{j+1})$ the vector of n_{j+1} ones, we can write the expression above as

$$\left(y_{n_{j+1}} - \begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ y_j(n_{j+1}) \odot h_j(n_{j+1})^T \end{bmatrix}^T \begin{bmatrix} \mu_{d_j} \\ \beta_j \end{bmatrix} \right).$$

Here we use $\odot$ to mean element wise multiplication so that we have

$$\begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ y_j(n_{j+1}) \odot h_j(n_{j+1})^T \end{bmatrix} = \begin{bmatrix} 1 & \cdots & 1 \\ y_j(x_1)h_j(x_1) & \cdots & y_j(x_{n_{j+1}})h_j(x_{n_{j+1}}) \end{bmatrix}$$

If we want to infer μ_{d_j} and β_j we need to infer the two at the same time.

Maximum likelihood estimation

Take the gradient of the likelihood with respect to the vector containing the parameters in μ_{d_j} and β_j. If we set this gradient to zero we have the vector of estimators

$$\left(\begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ y_j(n_{j+1}) \odot h_j(n_{j+1})^T \end{bmatrix} \Lambda_d(n_{j+1}) \begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ y_j(n_{j+1}) \odot h_j(n_{j+1})^T \end{bmatrix}^T \right)^{-1} \begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ y_j(n_{j+1}) \odot h_j(n_{j+1})^T \end{bmatrix} \Lambda_d(n_{j+1}) y_{n_{j+1}}.$$

The variance parameter $\sigma^2_{d_j}$ can be estimated using the usual formula. The correlation parameters θ_{d_j} are the argument that maximizes the likelihood after we plug-in the estimators for the rest of the parameters.

Bayesian model selection

In appendix A we write the prediction error for the case in which the mean parameter is unknown. We can derive an analogue formula for the multi-fidelity model above.

We look at the maximum likelihood estimator of μ_{d_j} and β_j as the random variable

$$\begin{bmatrix} M_{d_j} \\ B_j \end{bmatrix}$$

that is equal to

$$\left(\begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ Y_j(n_{j+1}) \odot h_j(n_{j+1}) \end{bmatrix} \Lambda_d(n_{j+1}) \begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ Y_j(n_{j+1}) \odot h_j(n_{j+1}) \end{bmatrix}^T \right)^{-1} \begin{bmatrix} \mathbf{1}(n_{j+1})^T \\ Y_j(n_{j+1}) \odot h_j(n_{j+1}) \end{bmatrix} \Lambda_d(n_{j+1}) Y_{j+1,n_{j+1}}.$$

This expression depends on two consecutive levels through $Y_j(n_{j+1})$ and $Y_{j+1,n_{j+1}}$. If we want to reproduce what is done for a single data set we can consider the conditional random variable

$$\begin{bmatrix} M_{d_j} \\ B_j \end{bmatrix} \Bigg| \{Y_j(n_{j+1}) = y_j(n_{j+1})\}$$

and use the usual priors described on appendix A. For the rest of the priors and a more in depth discussion the reader can consult [35] section 4.3.

Related literature

The model in which $Y_{2,x} = g(x)Y_{1,x} + Y_{d,x}$ has been studied by several authors. In [30] they use a constant function g.

Later Peter Quian in [43] developed a model where the scaling function $g(x)$ is a gaussian processes independent of Y_1 and Y_d. He uses a sequential Bayesian formulation where he uses MCMC methods.

Then Loic Le Gratiet in [35] gives a Bayesian formulation for all the parameters with the formulas for the universal kriging predictive distribution as well other formulas that reduce the complexity of the model.

The main difference between the models in [43] by P.Quian and [35] by L.Le Gratiet is that in the second he assumes the parametric form for the scaling functions $g_j(x) = \beta_j^T h_j(x)$. On the other hand in [43] the scaling function is estimated using a non-parametric Bayesian approach.

The main improvement in [35] is that we can separate the model into different parts and this allows faster inference and prediction strategies. This is true provided that the inputs sets of the data are nested.

So the two drawbacks of this method are the fact that we need to assume that $\mathcal{X}_N \subset \cdots \subset \mathcal{X}_2 \subset \mathcal{X}_1$ and that the scaling functions must be a linear combination of known functions. The other option is to use a G.P. as

a prior for the scaling function and have slower inference and prediction procedures.

In the next two sections we propose an Expectation Maximization algorithm to relax the nested design hypothesis and we derive a sequential strategy for a similar model with a scaling function that is a polynomial.

2.4 EM model selection for the case of disjoint set of observations

Assume that we sample two random processes, $Y_{1,x}$ and $Y_{2,x}$, indexed by the same set $\mathcal{X}$. Let $Y_{1,n_1} := \{Y_{1,x} : x \in \mathcal{X}_1\}$ and $Y_{2,n_2} := \{Y_{2,x} : x \in \mathcal{X}_2\}$ be the sets containing all the observed random variables. We suppose that $\mathcal{X}_1$ and $\mathcal{X}_2$ are finite and contain n_1 and n_2 indexes respectively.

We want to build a predictor using all the observed data Y_{1,n_1} and Y_{2,n_2}. If $Y_{2,x}$ is such that it can be written as $g(x)Y_{1,x} + Y_{d,x}$ where $\{Y_{d,x} : x \in \mathcal{X}\}$ and g is a known function. Then, when $\mathcal{X}_2$ is a subset of $\mathcal{X}_1$ the optimal predictor is

$$g(x)E[Y_{1,x}|Y_{n_1}] + E[Y_{d,x}|Y_d(n_2)].$$

Because $Y_d(n_2) := Y_{n_2} - g(n_2)Y_1(n_2)$, the equation above makes sense only when we observe $Y_1(n_2)$. In the next paragraphs, we apply an Expectation Maximization type algorithm to make predictions when $\mathcal{X}_1$ and $\mathcal{X}_2$ are disjoint. This algorithm keeps the sequential inference and prediction of the models discussed before.

An alternative solution was proposed in [35]. They begin by estimating the parameters of Y_1 and maximizing the likelihood of $y_{n_2} - g(n_2)E[Y_1(n_2)|Y_{n1} = y_{n_1}]$. We compare the two on a numerical example.

2.4.1 Expectation maximization

If we consider everything we said above, the setup is the following: we are given Y_{1,n_1} and Y_{2,n_2} and we want to estimate the parameters that define the distribution of $Y_{1,x}$ and $Y_{d,x}$. We do not specify the distributions for the time being.

We can estimate those of Y_{n_1}. On the other hand, we do not know $Y_d(n_2)$ so any direct data driven estimation is impossible. What is missing is $Y_1(n_2)$.

An Expectation Maximization or E.M. algorithm is an iterative procedure whose goal is to estimate the parameters by maximizing the likelihood when some data is missing.

To state the algorithm we simplify our usual notation and note $Y_{n_{12}}$ the vector containing Y_{1,n_1} and Y_{2,n_2} and ξ the vector of the parameters that define the distributions of Y_1 and Y_2. Let $q(z)$ be a pdf. Then,

$$\begin{aligned}
\log f_{Y_{n_{12}}}(y_{n_{12}};\xi) &= \cdots \\
\cdots &= \int \log\left(f_{Y_{n_{12}}}(y_{n_{12}};\xi)\right) q(z)\ dz \\
&= \int \log\left(f_{Y_{n_{12}}}(y_{n_{12}};\xi)\frac{f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi)}{f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi)}\frac{q(z)}{q(z)}\right) q(z)\ dz \\
&= \int \log\left(f_{Y_{n_{12}},Y_1(n_2)}(z,y_{n_{12}};\xi)\right) q(z)\ dz \\
&\quad - \int \log q(z)\ q(z)\ dz + \int \log\left(\frac{q(z)}{f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi)}\right) q(z)\ dz
\end{aligned}$$

The third term is the relative entropy between $q(z)$ and $f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi_{12})$. It is sometimes noted $D(q(z)||f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi_{12}))$. Because it is always

positive, we have that

$$\log f_{Y_{n_{12}}}(y_{n_{12}};\xi) \geq \int \log\left(f_{Y_1(n_2),Y_{n_{12}}}(z,y_{n_{12}};\xi)\right) q(z)\ dz - \int \log q(z)\ q(z)\ dz$$

The E.M. algorithm is based on this inequality. We do not know the true parameters so we start with an arbitrary guess that we note $\xi^{(0)}$. Then, we set $q(z)$ as the distribution of the unknown data, given the observations, but computed with the guessed parameters. This is

$$q(z) = f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi^{(0)}).$$

The inequality becomes

$$\begin{aligned}\log f_{Y_{n_{12}}}(y_{n_{12}};\xi) &\geq \cdots \\ \cdots &\geq \int f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi^{(0)}) \log\left(f_{Y_1(n_2),Y_{n_{12}}}(z,y_{n_{12}};\xi)\right) dz \\ &\quad - \int f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi^{(0)}) \log\left(f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi^{(0)})\right) dz \\ &=: Q(\xi,\xi^{(0)}) + G(\xi^{(0)}).\end{aligned}$$

In particular, because $D(q(z)||q(z)) = 0$, we have that when $\xi^{(0)} = \xi$, the first inequality above is an equality so that $\log f_{Y_{n_{12}}}(y_{n_{12}};\xi) = Q(\xi,\xi) + G(\xi)$.

The algorithm computes the expectation, $Q(\xi,\xi^{(0)})$, on the right and sets our new guess as a value that maximize it. By running the algorithm, we obtain a sequence of estimates, $\xi^{(0)}, \xi^{(1)}, \ldots$, such that

$$\log f_{Y_{n_{12}}}(y_{n_{12}};\xi^{(0)}) \leq \log f_{Y_{n_{12}}}(y_{n_{12}};\xi^{(1)}) \leq \cdots$$

This is because

$$\begin{aligned}\log f_{Y_{n_{12}}}(y_{n_{12}};\xi^{(t)}) &= Q(\xi^{(t)},\xi^{(t)}) + G(\xi^{(t)}) \\ &\leq \max_{\xi}\Big(Q(\xi,\xi^{(t)})\Big) + G(\xi^{(t)}) \\ &= Q(\xi^{(t+1)},\xi^{(t)}) + G(\xi^{(t)}) \\ &\leq \log f_{Y_{n_{12}}}(y_{n_{12}};\xi^{(t+1)}).\end{aligned}$$

This algorithm is only guaranteed to find a local maximum. To summarize

Algorithm 1 EM. algorithm

1: **procedure** EM
2: Guess the parameters $\xi^{(0)}$.
3: **while** $f_{Y_{n_{12}}}(y_{n_{12}};\xi^{(t-1)}) < f_{Y_{n_{12}}}(y_{n_{12}};\xi^{(t)})$ **do** ▷ Increasing Lik.
4: Compute $Q(\xi,\xi^{(t-1)})$.
5: Set $\xi^{(t)}$ as a value that maximizes $Q(\xi,\xi^{(t-1)})$.

E-step

So far we have not used any hypothesis about the relationship between the two observed processes. Consider the model in which

$$Y_{2,x} = g(x)Y_{1,x} + Y_{d,x}$$

where $Y_{d,x'}$ is independent of $Y_{1,x}$ for any x and x' and $g(x)$ is a known function.

We also assume that the processes we were working with are Gaussian and defined by some parameters as follows:

$$\begin{aligned}Y_{1,x} &\sim GP(\mu_1,\ \sigma_1^2\gamma_1(x,x';\theta_1)) \\ Y_{d,x} &\sim GP(\mu_d,\ \sigma_d^2\gamma_d(x,x';\theta_d)).\end{aligned}$$

By the relationship, the remaining process is

$$Y_{2,x} \sim GP\left(g(x)\mu_1 + \mu_d,\ g(x)\sigma_1^2\gamma_1(x,x';\theta_1)g(x') + \sigma_d^2\gamma_d(x,x';\theta_d)\right).$$

We continue with our usual notation and write $\Gamma(x,x') := \sigma^2\gamma(x,x';\theta)$, $\lambda(n;\theta) := \gamma(n;\theta)^{-1}$ and $\Lambda(n) := \Gamma(n)^{-1}$.
Let ξ_1 and ξ_d be the vectors containing all the parameters of Y_1 and Y_d. Sometimes instead of writing the two vectors we write ξ_{1d} for short. Then, the expectation we are interested in is

$$\begin{aligned}
Q(\xi_{1d},\xi_{1d}^{(t-1)}) &= \int \log\left(f_{Y_1(n_2),Y_{n_{12}}}(z,y_{n_{12}};\xi_{1d})\right) f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi_{1d}^{(t-1)})dz \\
&= \int \log\left(f_{Y_1(n_2),Y_{1,n_1}}(z,y_{n_1};\xi_1)\right) f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi_{12}^{(t-1)})dz \\
&+ \int \log\left(f_{Y_d(n_2)}\left(y_{n_2} - g(n_2)\odot z;\xi_d\right)\right) f_{Y_1(n_2)|Y_{n_{12}}}(z|y_{n_{12}};\xi_{1d}^{(t-1)})dz \\
&= E_{Y_1(n_2)}\left[\log\left(f_{Y_1(n_2),Y_{1,n_1}}(Y_1(n_2),Y_{1,n_1};\xi_1)\right)|Y_{n_{12}};\xi_{12}^{(t-1)}\right] \\
&+ E_{Y_1(n_2)}\left[\log\left(f_{Y_d(n_2)}(Y_{2,n_2} - g(n_2)\odot Y_1(n_2);\xi_d)\right)|Y_{n_{12}};\xi_{1d}^{(t-1)}\right] \\
&=: Q(\xi_1,\xi_{1d}^{(t-1)}) + Q(\xi_d,\xi_{1d}^{(t-1)}).
\end{aligned}$$

Before moving onto the $E-step$ computations notice the following. We could have argued that the missing values were $Y_d(n_2)$ instead of $Y_1(n_2)$. Then, we would have had

$$\begin{aligned}
Q(\xi_{1d},&\xi_{1d}^{(t-1)}) = \cdots \\
\cdots &= E_{Y_d(n_2)}\left[\log\left(f_{Y_1(n_2),Y_{1,n_1}}\left(\frac{1}{g(n_2)}\odot(Y_{2,n_2} - Y_d(n_2)),Y_{1,n_1};\xi_1\right)\right)|Y_{n_{12}};\xi_{1d}^{(t-1)}\right] \\
&+ E_{Y_d(n_2)}\left[\log\left(f_{Y_d(n_2)}(Y_d(n_2);\xi_d)\right)|Y_{2,n_2};\xi_{1d}^{(t-1)}\right].
\end{aligned}$$

We work with the expectations with respect to $Y_1(n_2)$. Using the formula for the expectation of a quadratic form we have that

$$\begin{aligned}
Q(\xi_1,&\xi_{1d}^{(t-1)}) = -\frac{n_{12}}{2}\log 2\pi - \frac{1}{2}\log|\Gamma_1(n_{12})| - \frac{1}{2}tr\left(\Lambda_1(n_2,n_2)\Gamma_{1|12}(n_2;\xi_1^{(t-1)})\right) \\
&- \frac{1}{2}\begin{bmatrix} Y_{1,n_1} - \mu_1\mathbf{1}(n_1) \\ \mu_{1|12}(n_2;\xi_{1d}^{(t-1)}) - \mu_1\mathbf{1}(n_2)\end{bmatrix}^T \Lambda_1(n_{12}) \begin{bmatrix} Y_{1,n_1} - \mu_1\mathbf{1}(n_1) \\ \mu_{1|12}(n_2;\xi_{1d}^{(t-1)}) - \mu_1\mathbf{1}(n_2)\end{bmatrix}
\end{aligned}$$

where

$$\begin{aligned}
\Lambda_1(n_2, n_2) &:= (\Gamma_1(n_2) - \Gamma_1(n_2, n_1)\Lambda_1(n_1)\Gamma_1(n_1, n_2))^{-1} \\
\Gamma_{1|12}(n_2; \xi_{1d}^{(t-1)}) &:= Cov(Y_1(n_2), Y_1(n_2)|Y_{n_{12}}; \xi_{1d}^{(t-1)}) \\
\mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) &:= E[Y_1(n_2)|Y_{n_{12}}; \xi_{1d}^{(t-1)}]
\end{aligned}$$

The second term is

$$\begin{aligned}
&Q(\xi_d, \xi_{1d}^{(t-1)}) = \cdots \\
&\cdots - \frac{1}{2}\log 2\pi - \frac{1}{2}\log|\Gamma_d(n_2)| - \frac{1}{2}tr\left(\Lambda_d(n_2)\left(g(n_2)g(n_2)^T \odot \Gamma_{1|12}(n_2; \xi_{1d}^{(t-1)})\right)\right) \\
&\quad - \frac{1}{2}(Y_{n_2} - g(n_2) \odot \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) - \mu_d \mathbf{1}(n_2))^T \Lambda_d(n_2) \cdots \\
&\qquad\qquad \cdots (Y_{2,n_2} - g(n_2) \odot \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) - \mu_d \mathbf{1}(n_2)).
\end{aligned}$$

M-step

Now we maximize $Q(\xi_1, \xi_{1d}^{(t-1)})$ and $Q(\xi_d, \xi_{1d}^{(t-1)})$ over ξ_1 and ξ_d.

Since the objective functions look almost the same as the likelihood of the observed vectors, we do as with the maximum likelihood estimation of the parameters. The parameter that maximizes the functions are

$$\widehat{\mu}_1 := \left(\mathbf{1}(n_{12})^T \lambda_1(n_{12}; \theta_1)\mathbf{1}(n_{12})\right)^{-1}\left(\mathbf{1}(n_{12})^T \lambda_1(n_{12}; \theta_1)\begin{bmatrix} Y_{n_1} \\ \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) \end{bmatrix}\right)$$

and

$$\widehat{\mu}_d := \left(\mathbf{1}(n_2)^T \lambda_d(n_2; \theta_d)\mathbf{1}(n_2)\right)^{-1}\left(\mathbf{1}(n_2)^T \lambda_d(n_2; \theta_d)(Y_{n_2} - g(n_2) \odot \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}))\right).$$

The estimators for the variance parameters are

$$\begin{aligned}
\widehat{\sigma_1}^2 &:= \frac{1}{n_{12}}\begin{bmatrix} Y_{n_1} - \widehat{\mu}_1\mathbf{1}(n_1) \\ \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) - \widehat{\mu}_1\mathbf{1}(n_2) \end{bmatrix}^T \lambda_1(n_{12}; \theta_1)\begin{bmatrix} Y_{n_1} - \widehat{\mu}_1\mathbf{1}(n_1) \\ \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) - \widehat{\mu}_1\mathbf{1}(n_2) \end{bmatrix} \\
&\quad + \frac{1}{n_{12}}tr\left(\lambda_1(n_2, n_2; \theta_1)\Gamma_{1|12}(n_2; \xi_1^{(t-1)})\right)
\end{aligned}$$

and

$$\widehat{\sigma_d}^2 := \frac{1}{n_2}(Y_{n_2} - g(n_2) \odot \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) - \widehat{\mu}_d(\theta_d)\mathbf{1}(n_2))^T \lambda_d(n_2; \theta_d) \cdots$$
$$\cdots (Y_{n_2} - g(n_2) \odot \mu_{1|12}(n_2; \xi_{1d}^{(t-1)}) - \widehat{\mu}_d \mathbf{1}(n_2))$$
$$+ \frac{1}{n_2} tr\left(\lambda_d(n_2; \theta_d)\left(g(n_2)g(n_2)^T \odot \Gamma_{1|12}(n_2; \xi_{1d}^{(t-1)})\right)\right).$$

We plug all the estimators above in the objective functions to obtain two expressions that only depend on the θ's. The estimator for these parameters are defined as solutions to

$$\underset{\theta_1}{\text{maximize}} \quad -\frac{n_{12}}{2}\log \widehat{\sigma_1}^2(\theta_1) - \frac{1}{2}\log|\gamma_1(n_{12}; \theta_1)|$$

and

$$\underset{\theta_d}{\text{maximize}} \quad -\frac{n_2}{2}\log \widehat{\sigma_d}^2(\theta_d) - \frac{1}{2}\log|\gamma_d(n_2; \theta_d)|.$$

Finally we update our parameters by using the rules below.

$$\theta^{(t)} := \widehat{\theta},$$
$$\mu^{(t)} := \widehat{\mu}(\theta^{(t)}),$$
$$\sigma^{2(t)} := \widehat{\sigma}^2(\theta^{(t)}).$$

The algorithm above can be extended to the case in which $\mathcal{X}_1 \cap \mathcal{X}_2 = \mathcal{X}_{1\cap 2}$. The function $Q(\xi_1, \xi_{1d}^{(t-1)})$ and the estimators for the parameters of Y_1 are the same. The part corresponding to the difference process now has an extra term.

$$Q(\xi_d, \xi_{1d}^{(t-1)}) = -\frac{1}{2}\log 2\pi - \frac{1}{2}\log|\Gamma_d(n_2)|$$
$$- \frac{1}{2}\left(\widehat{Y}_d(n_2) - \mu_d\mathbf{1}(n_2)\right)^T \Lambda_1(n_2)\left(\widehat{Y}_d(n_2) - \mu_d\mathbf{1}(n_2)\right)$$
$$- \frac{1}{2} tr\left(\lambda_d(n_{2\setminus 1\cap 2}; \theta_d)\left(g(n_{2\setminus 1\cap 2})g(n_{2\setminus 1\cap 2})^T \odot \Gamma_{1|12}(n_{2\setminus 1\cap 2}; \xi_{1d}^{(t-1)})\right)\right)$$

where

$$\widehat{Y}_d(n_2) := \begin{bmatrix} Y_d(n_{1\cap 2}) \\ Y_2(n_{2\setminus 1\cap 2}) - g(n_{2\setminus 1\cap 2})\mu_{1|12}(n_{2\setminus 1\cap 2}; \xi_{1d}^{(t-1)}) \end{bmatrix}.$$

For the estimators we replace $Y_{n_2} - g(n_2)\mu_{1|12}(n_{1\cap 2}; \xi_{1d}^{(t-1)})$ with $\widehat{Y}_d(n_2)$ in the formulas above.

Numerical example

We test the method on a simulated example. For some fixed parameters we sample

$$Y_{1,x} \sim GP(\mu_1,\ \sigma_1^2\gamma_1(x, x'; \theta_1))$$
$$Y_{d,x} \sim GP(\mu_d,\ \sigma_d^2\gamma_d(x, x'; \theta_d)).$$

independently over two disjoint input sets $\mathcal{X}_1$ and $\mathcal{X}_2$, to get $y_d(n_2)$, $y_1(n_2)$ and y_{1,n_1}. We fix a constant β and using the formula

$$Y_{2,x} = \beta Y_{1,x} + Y_{d,x}$$

we compute the sample $y_{2,n_2} := \beta y_1(n_2) + y_d(n_2)$. We do the same to build a test set $y_{2,n_{test}}$.

The objective is to predict Y_2 at the test set and to recover the parameters of Y_1 and Y_d along β, by using the simulated data, or training sets, $(\mathcal{X}_1, y_{1,n_1})$ and $(\mathcal{X}_2, y_{2,n_2})$.

The inputs x are in $[0, 4]$ and the two correlation functions are Matern 3/2. The true parameters are fixed to

$\mu_1 = 0.0$	$\sigma_1 = 0.01$	$\theta_1 = 1$	$\Vert$	$\mu_d = 0.0$	σ_d =0.04	$\theta_d = 0.8$	β =2

Table 2.1: True parameters.

The paths of Y_1 do not vary as much as those of Y_d. Most of the variation

of Y_2 is due to the difference process as shown on the plot 2.3.

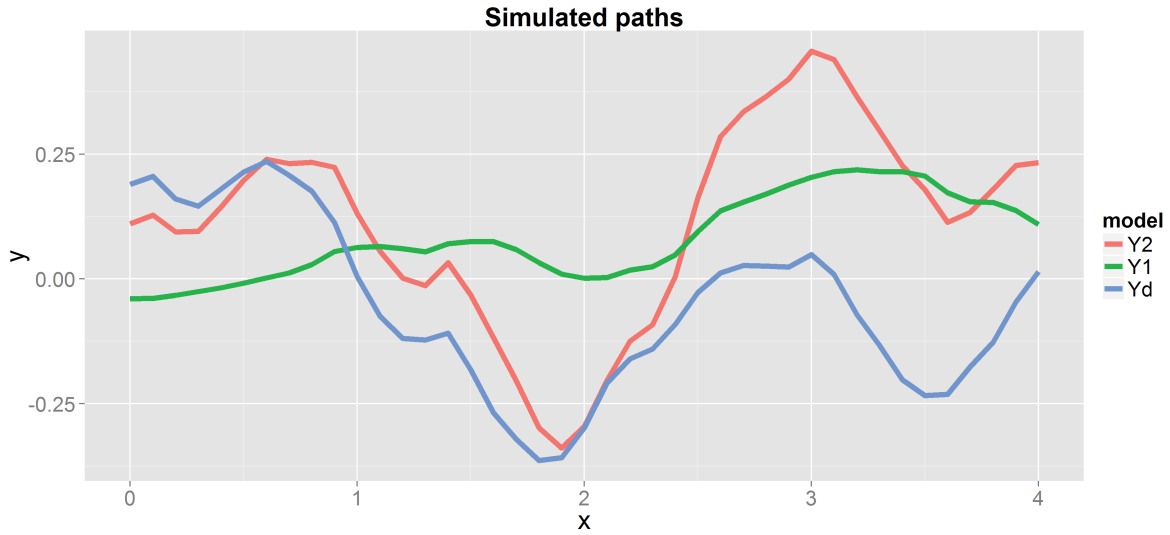


Figure 2.3: Three simulated paths using the parameters on table 2.1.

We use a decreasing size training sets and test sets, see 2.4. For each size we simulate 100 different training sets and for each training set we compute the $RMSE$ of the prediction at the test set.

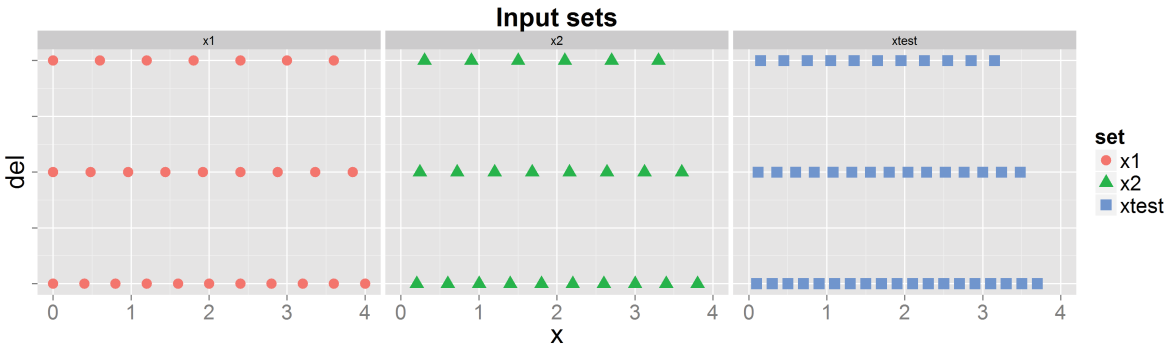


Figure 2.4: Each row represents the combination of input set used in the example. For the bottom row we have $(n_1, n_2, n_{test}) = (11, 10, 19)$. The middle and top rows have $(9, 8, 15)$ and $(7, 6, 11)$.

We compare the EM algorithm to a procedure we note K. For K we learn the parameters of Y_1 and Y_2 using their corresponding training sets. Then, we predict $y_1(n_2)$ with $\widehat{y}_1(n_2) := E[Y_1(n_2)|Y_{n_1} = y_{n_1}]$ and use $y_d(n_2) := y_{n_2} - \beta_{true}\ \widehat{y}_1(n_2)$ to estimate the parameters of the difference process.

The predicted values for K we consider are $\widehat{y}_1(n_{test})$ for the first process

Y_1; $\widehat{y}_d(n_{test}) := E[Y_d(n_{test})|Y_d(n_2) = y_d(n_2)]$ for the difference process. For the target process we use $\widehat{y}_2(n_{test}) := E[Y_2(n_{test})|Y_{2,n_2} = y_{2,n_2}]$.

In the boxplots below show the results of the simulations. In each column the boxplot of the $RMSE$ over 100 simulations is shown. They are ordered from left to right for decreasing number of training points.

The predictions for $y_1(n_{test})$ in 2.5a are equivalent although the parameters are estimated using different formulas. This is despite the fact that in EM we estimates the scaling parameter β too, whereas for K it is fixed to the true value.

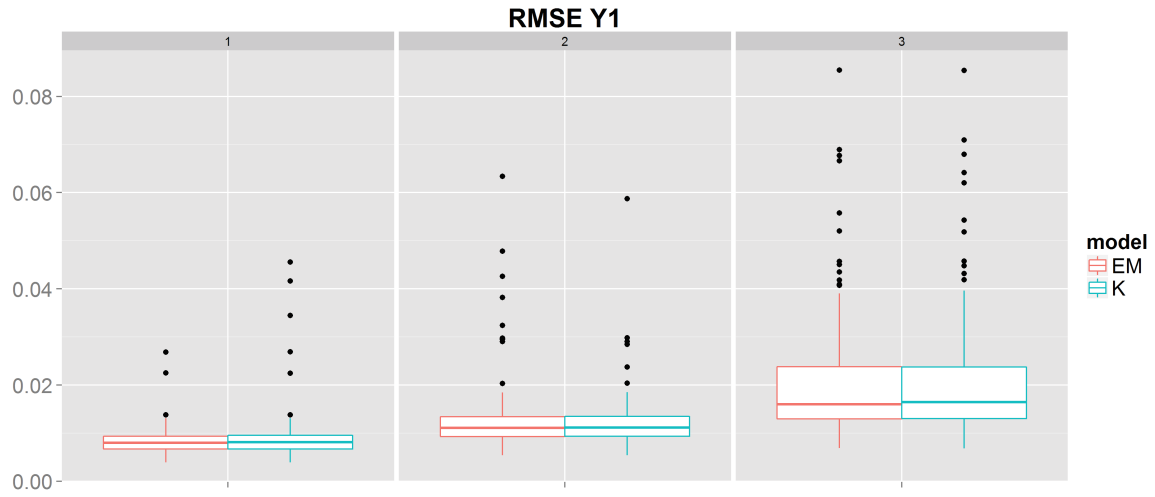
The estimation is visibly better when it comes to the predictions of the difference process Y_d as shown in 2.5b. This means that we expect to be able to use the observations of Y_1 to help us in the prediction of the target process. Figure 2.5c confirms this: the error is significantly smaller than the one we obtain by using $(\mathcal{X}_2, y_{2,n_2})$ only.

So although the difference between the processes is big, we can incorporate the data of Y_1 to decrease the prediction error.

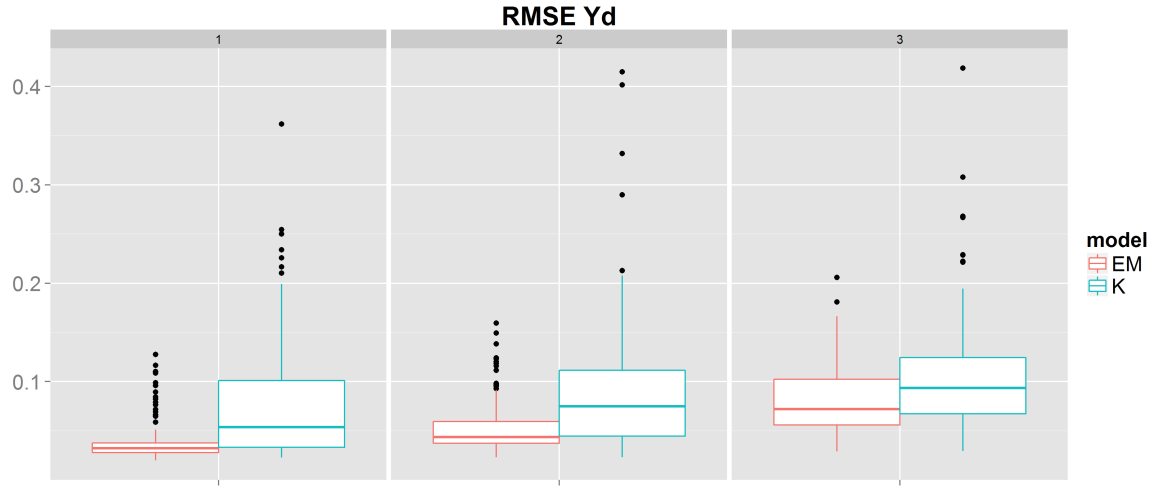
The results concerning the predicted parameters are shown on figure 2.6. The mean and variance parameters are accurately estimated by both methods.

Because the predictions of the first process are accurate, the data $y_d(n_2) := y_{n_2} - \beta_{true}\ \widehat{y}_1(n_2)$ used to train the difference process is close to the true too and so the estimation of the covariance parameter of the Y_d for K is better than for EM.

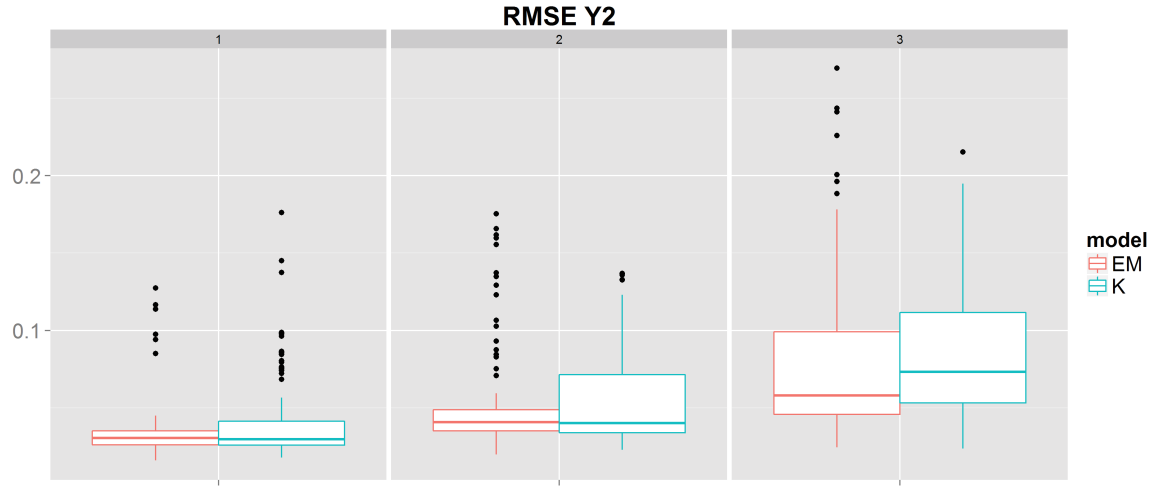
The predicted scaling parameter β has a very big variance as shown in figure 2.6. It seems that the EM estimation attributes the variability of the data to the scaling parameter instead of the variance or covariance parameters.



(a) $RMSE$ of the predictor for $y_1(n_{test})$. The two models have equivalent results. Both give very good results.



(b) EM predicts $y_d(n_{test})$ more accurately.



(c) $RMSE$ of the predictor for $y_2(n_{test})$.

Figure 2.5: Boxplots of the $RMSE$ of prediction. Decreasing number of training points from left to right as shown in figure 2.4.

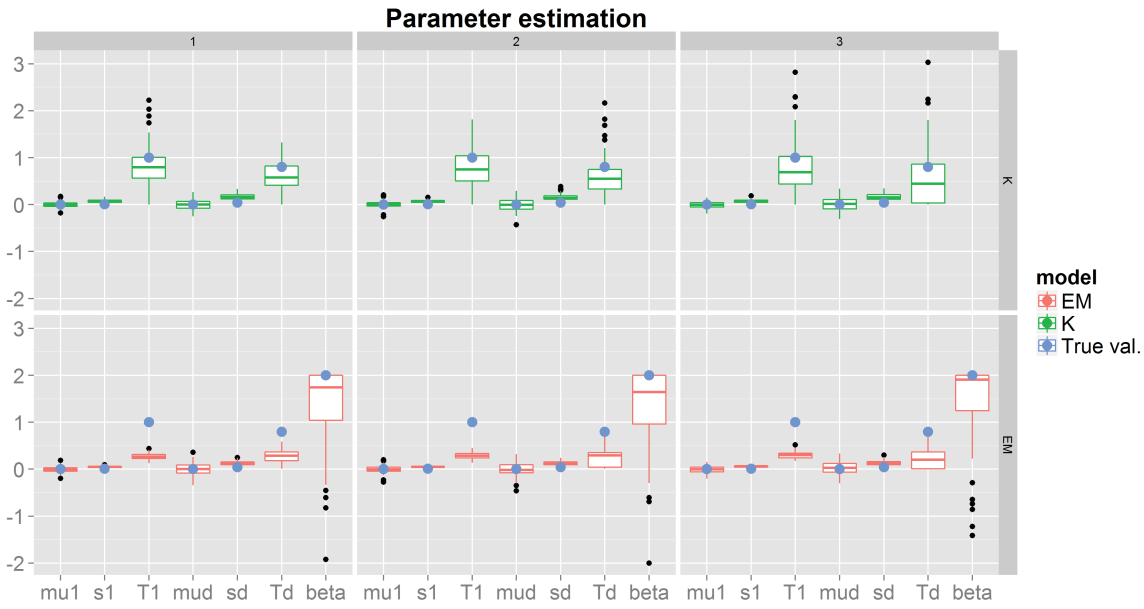


Figure 2.6: Boxplots of estimated parameters for decreasing number of training points as in 2.4.

2.5 Multi-fidelity regression with polynomial relationships

Here we extend the sequential multi-fidelity model of the previous sections to the case in which the scaling function is a polynomial $P(y) = \sum_{j=1}^{N} y^j a_j(x)$. The way we showed how to decompose the predictive mean and variance as well as the likelihood of the data is going to be useful here.

For simplicity we consider only two data sets $(\mathcal{X}_1, Y_{1,n_1})$ and $(\mathcal{X}_2, Y_{2,n_2})$ drawn from two random processes $\{Y_{1,x} : x \in \mathcal{X}\}$ and $\{Y_{2,x} : x \in \mathcal{X}\}$.

We assume that there is a third process $\{Y_{d,x} : x \in \mathcal{X}\}$ independent of Y_1 such that we can write

$$Y_{2,x} = P(Y_{1,x}) + Y_{d,x}.$$

Where $P(y) = \sum_{j=1}^{N} a_j y^j$ is a polynomial with real fixed coefficients. For the time being we do not assume any particular distribution for the processes involved.

Proposition 3. *If we observe $Y_1(n_2)$ in Y_{1,n_1}, then the distribution of the observations at y_{n_1}, y_{n_2} is*

$$f_{Y_{1,n_1},Y_{2,n_2}}(y_{n_1}, y_{n_2}) = f_{Y_{1,n_1}}(y_{n_1}) f_{Y_d(n_2)}(y_d(n_2))$$

where $y_d(n_2) := y_{n_2} - P(y_1(n_2))$. Also, the optimal predictor $\widehat{Y}_{2,x}$ at an observed index x for $Y_{2,x}$ given the observations is

$$E[Y_{2,x}|Y_{1,n_1}, Y_{2,n_2}] = E[P(Y_{1,x})|Y_{1,n_1}] + E[Y_{d,x}|Y_d(n_2)].$$

with prediction error

$$E[(Y_{2,x} - \widehat{Y}_{2,x})^2|Y_{1,n_1}, Y_{2,n_2}] = Var(P(Y_{1,x})|Y_{1,n_1}) + Var(Y_{d,x}|Y_d(n_2)).$$

Proof. We can get the decomposition of the density by making the change of variables

$$f_{Y_{1,n_1},Y_{2,n_2}}(y_{n_1}, y_{n_2}) = f_{Y_{1,n_1},Y_d(n_2)}(y_{n_1}, y_d(n_2)) \left| \frac{\partial(y_{n_1}, y_d(n_2))}{\partial(y_{n_1}, y_{n_2})} \right|.$$

Where the absolute value of the determinant of the Jacobian is 1. Then, we use the independence between Y_1 and Y_d.

For the conditional expectation we first write

$$E[Y_{2,x}|Y_{n_1}, Y_{n_2}] = E[P(Y_{1,x})|Y_{n_1}, Y_{n_2}] + E[Y_{d,x}|Y_{n_1}, Y_{n_2}].$$

The second term is already familiar to us and it is equal to $E[Y_{d,x}|Y_d(n_2)]$. For the first term we use the Law of the Unconscious Statistician or LOTUS and write

$$\begin{aligned} E[P(Y_{1,x})|Y_{1,n_1}, Y_{2,n_2}] &= \int P(s) f_{Y_{1,x}|Y_{1,n_1},Y_{2,n_2}}(s|y_{n_1}, y_{n_2}) ds \\ &= \int P(s) f_{Y_{1,x}|Y_{1,n_1}}(s|y_{n_1}) ds \\ &= E[P(Y_{1,x})|Y_{1,n_1}]. \end{aligned}$$

For the prediction error, we can expand $E[(Y_{2,x} - \widehat{Y}_{2,x})^2|Y_{1,n_1}, Y_{2,n_2}]$ to get

the sum of three terms

$$E[(P(Y_{1,x}) - \widehat{P}(Y_{1,x}))^2 | Y_{1,n_1}, Y_{2,n_2}] + E[(Y_{d,x} - \widehat{Y}_{d,x})^2 | Y_{1,n_1}, Y_{2,n_2}] \\ + 2E[(P(Y_{1,x}) - \widehat{P}(Y_{1,x}))(Y_{d,x} - \widehat{Y}_{d,x}) | Y_{1,n_1}, Y_{2,n_2}].$$

Where $E[P(Y_{1,x})|Y_{1,n_1}]$ and $E[Y_{d,x}|Y_d(n_2)]$ are noted as $\widehat{P}(Y_{1,x})$ and $\widehat{Y}_{d,x}$. Then, $\widehat{P}(Y_{1,x})$ is a function of Y_{1,n_1} and $\widehat{Y}_{d,x}$ is a function of $Y_d(n_2)$.

Take a look at the first expectation. Given Y_{1,n_1}, $(P(Y_{1,x}) - \widehat{P}(Y_{1,x}))^2$ is a function of $Y_{1,x}$. Then,

$$\begin{aligned} E[(P(Y_{1,x}) - \widehat{P}(Y_{1,x}))^2 | Y_{1,n_1}, Y_{2,n_2}] &= E[(P(Y_{1,x}) - \widehat{P}(Y_{1,x}))^2 | Y_{1,n_1}] \\ &= Var(P(Y_{1,x})|Y_{n_1}). \end{aligned}$$

Using a similar argument, we have that the second expectation is the conditional variance $Var(Y_{d,x}|Y_d(n_2))$.

The crossed term is the conditional covariance so it is equal to

$$E[P(Y_{1,x})Y_{d,x}|Y_{1,n_1}, Y_{2,n_2}] - \widehat{P}(Y_{1,x})\widehat{Y}_{d,x}.$$

Using LOTUS on the first term we get

$$\begin{aligned} E[P(Y_{1,x})Y_{d,x}|Y_{1,n_1}, Y_{2,n_2}] &= \int\int P(s)t f_{Y_{1,x},Y_{d,x}|Y_{1,n_1},Y_{2,n_2}}(s,t|y_{n_1}, y_{n_2})dsdt \\ &= \left(\int P(s) f_{Y_{1,x}|Y_{1,n_1}}(s|y_{n_1})ds\right) \\ &\quad \left(\int t f_{Y_{d,x}|Y_d(n_2)}(t|y_d(n_2))dt\right) \\ &= \widehat{P}(Y_{1,x})\widehat{Y}_{d,x}. \end{aligned}$$

It turns out that the two processes at x, conditionally on the observations, are uncorrelated too. The prediction error is

$$Var(P(Y_{1,x})|Y_{1,n_1}) + Var(Y_{d,x}|Y_d(n_2)).$$

□

In the proposition above we considered a polynomial with constant coefficients but we can replace any coefficient by a function of the input x. In this sense, this model is a generalization of the additive model we described above.

Gaussian processes case

Because P is a polynomial, the prediction and prediction error depend on the moments of $Y_{1,x}|Y_{1,n_1}$ and $Y_{d,x}|Y_d(n_2)$. If we assume that

$$Y_{1,x} \sim GP(\mu_1, \Gamma_1(x, x'))$$
$$Y_{d,x} \sim GP(\mu_d, \Gamma_d(x, x'))$$

then, we know the conditional distributions and also all of its moments.

Proposition 3 also shows how the density function of the observations factors. We can learn the parameters of Y_1 and Y_d independently by using the maximum likelihood estimators.

As before, we can estimate the parameters that define P. If we note $P(y) = \sum_{j=1}^{N} y^j a_j$ as $p(y)^T a$, the maximum likelihood estimation for the vector containing μ_d and a, given the observations, is

$$\left(\begin{bmatrix} \mathbf{1}(n_2)^T \\ p(y_1(n_2))^T \end{bmatrix} \Gamma(Y_d(n_2))^{-1} \begin{bmatrix} \mathbf{1}(n_2)^T \\ p(y_1(n_2))^T \end{bmatrix}^T \right)^{-1} \begin{bmatrix} \mathbf{1}(n_2)^T \\ p(y_1(n_2))^T \end{bmatrix} \Lambda_d(n_2) y_{n_2}.$$

where $p(y_1(n_2))$ is the vector with entries $p(y_{1,x_j})$ for all the indexes x_j of $Y_1(n_2)$.

For the Bayesian inference we can use the same priors as with the linear model. This is we assume a prior distribution on μ_d and a, given $p(y_1(n_2))$. Inference regarding the parameters of Y_1 is exactly the same as for the linear model. For the details on the a priori distributions see [35].

This opens the door door to more complex parametric models.

2.6 Conclusion

Gaussian processes regression. In this chapter, we described a way to use Gaussian processes to model unknown functions. We made an a priori assumption that involves a parametric form for the mean and covariance of the process. We inferred these parameters and built a predictive distribution.

Multi-fidelity with Gaussian processes. This method can be extended to the case in which we dispose of several data sets that are related to the target unknown function. We suppose we are given an order of fidelity for these data sets and in this case, the key idea is to assume that for each data set there is a difference processes that is independent of the observed process. By using this independence structure we can make a sequential prediction in which we learn the parameters of all the observed and difference processes except for those of the target process.

Disjoint input sets. This procedure depends on the fact that the observations are made over nested input sets. We proposed an EM type algorithm to estimate the parameters and compute the prediction function for the case of disjoint sets.

Polynomial relationship between successive fidelity levels. We showed that the independence structure of the linear multi-fidelity model can be extended to almost any parametric relationship. In particular we wrote the formulas for a polynomial relationship and argued that in the case of Gaussian processes these formulas can be computed analytically. The inference in these models is analogous to the linear case but more complex.

We end the chapter with a very flexible model that is also very complex as an a priori assumption. In the next chapter we infer the relationship between the processes using a nonparametric technique.

Chapter 3

Nonparametric Model

In the previous chapter we described the problem of multi-fidelity regression. We want to model data that is difficult to observe. This data can be the result of an experiment or a computer simulation.

In multi-fidelity we assume that we dispose of simplified experiments or simulations for which we can obtain data more easily and the goal is to use this simplified data, along the one that is hard to obtain to enhance our understanding of the complex experiment.

We model this setup by assuming that we have N data sets, that we note $(\mathcal{X}_1, y_{1,n_1}), ..., (\mathcal{X}_N, y_{N,n_N})$, generated from N different random processes $Y_1, ..., Y_N$.

Following the articles [30, 43, 35], we assume that we are given an order of fidelity. More precisely, we have that for each process Y_j there is an independent difference process Y_{d_j} and a function g_j such that

$$Y_{j+1,x} = g_j(x)Y_{j,x} + Y_{d_j,x}. \tag{3.1}$$

This equation models the relationship between the data sets. We extended this model to one in which the independence structure is identical, but the

relationship between two successive levels is

$$Y_{j+1,x} = P(Y_{j,x}) + Y_{d_j,x} \tag{3.2}$$

where $P(y) := \sum_{j=1}^{K} a_j(x) y^j$. Because of the independence structure we showed that it is possible to make inference and predictions sequentially as done in [35].

But we never really questioned the form in which the processes are related. Although the models built using equations (3.1) and (3.2) can be very flexible it is not clear why should we assume any particular parametric form for the relationship.

In this chapter we will explore the effect that assuming a particular relationship has on prediction and propose a model in which we estimate this relationship nonparametrically.

We start with some examples for which the relationship between the data is nonlinear. Then, we introduce a multi-fidelity model based on Gaussian processes where the relationship is unknown. We propose a predictor with its predictor error and we return to the study of two numerical examples.

3.1 Unknown relationship

In this section we present a multi-fidelity model based on Gaussian processes. As before we assume that we know the order in which the processes are related but instead of assuming a parametric form for the relationship we will consider that

$$Y_{j+1,x} = \varphi(Y_{j,x}) + Y_{d_j,x}$$

where φ is an unknown function. Before describing the model in detail we introduce an example that motivates this type of model.

A numerical example

Lets consider the following example taken from page 240 of the book by Selvadurai [48]. We solve

$$a^2 \nabla^2 p(x,t) = \frac{\partial p}{\partial t}(x,t) \tag{3.3}$$

in a rectangular domain Ω. We will extract the fluid through the top of the domain at a constant rate and consider that the walls and floor are impermeable.

This is a diffusion type partial differential equation that describes the variation of pressure of a liquid in a porous fabric. It takes into account the compressibility of the fluid and the fabric through the expression of a^2 given by

$$a^2 := \frac{K}{\gamma\,(n^* C_f + C_s)}.$$

We will solve for the pressure p and consider (K, n^*), two of the parameters that define a^2, as our inputs. The first parameter, K, is the hydraulic conductivity and n^* is the porosity of the porous fabric. The other parameters are: γ, the unit weight of the fluid; and C_s and C_f the compressibility of the fabric and the fluid.

For a fixed (K, n^*) and a given mesh size N, we solve for p numerically over all of the domain using freefem++ [36]. We will consider the maximum pressure of the numerical solution after a fixed time on the domain as the output. We note it $p(K, n^*; N)$.

We assume that the inputs vary in closed intervals continuously. The limits were taken from an online database [20]. The other parameters are

fixed as shown on the list below.

$$K \in [1e-12, 1], \quad n^* \in [0.10, 0.85],$$
$$\gamma = 6.67, \quad C_f = 1.06, \quad C_s = 2e-10.$$

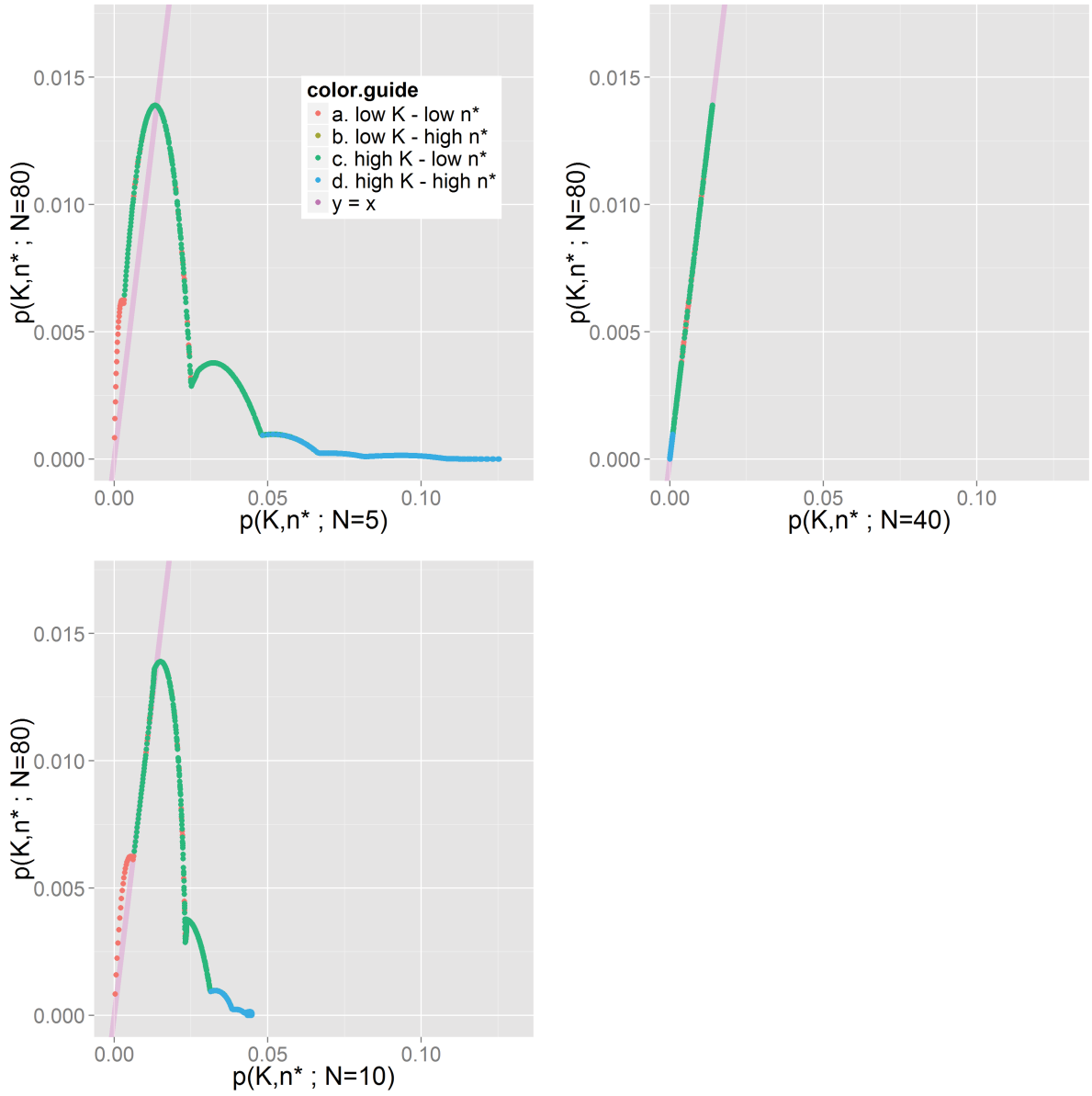


Figure 3.1: Plots of pairs of the form $(p(K, n^*; N), p(K, n^*, 80))$. The solid line is the identity function $y = x$. Each color represents a class of pairs (K, n^*).

The fidelity of the outputs will be indexed by N: a bigger N defines a more complex numerical model and a more precise response.

Figure 3.1 shows the relationship between $p(K, n^*; N)$ and $p(K, n^*; 80)$ for different values of N. In the last panel we plotted the output for $N = 40$. We see that the numerical model converged. We will assume that $N = 40$ is the highest fidelity level.

The other panels show relationships that are very hard to describe using a parametric form. In the next section we address the problem of estimating a relationship like the one on the first column of figure 3.1 by using locally linear polynomials.

3.1.1 Nonparametric estimation

In this section, for ease of notation, we study the case of two data sets, $(\mathcal{X}_1, y_{1,n_1})$ and $(\mathcal{X}_2, y_{2,n_2})$, generated by two processes Y_1 and Y_2. We assume that $\mathcal{X}_2 \subset \mathcal{X}_1$ and that there is function φ and a difference process Y_d such that

$$Y_{2,x} = \varphi(Y_{1,x}) + Y_{d,x}.$$

Our first objective is to estimate φ, the relationship between Y_1 and Y_2, by using the training set $(y_1(n_2), y_{2,n_2})$ where $y_1(n_2)$ are the observations of Y_1 on $\mathcal{X}_2$.

Then, we discuss the problem of building a predictor with its prediction error for $Y_{2,x}$ at an unobserved input x by using the estimated relationship $\widehat{\varphi}$.

By using this model, we do not make any additional assumptions about φ and we avoid adding any new parameters. This is specially convenient for inputs whose dimension D is bigger than 1. The usual alternative - see [30, 43, 35] - is to set $\varphi(y) = (\beta_0 + \beta_1^T x)y$ which means $D + 1$ parameters when $x \in \mathbf{R}^D$. Estimating the relationship will be a 1 dimensional problem independently of x. We do not need additional data to estimate β_0 and β_1 as D grows.

We will be able to describe relationships similar to the ones in the first column of figure 3.1.

Locally linear polynomials

We are given data in the form of pairs $(y_1(n_2), y_{2,n_2})$ and we want to predict $Y_{2,x}$ at $Y_{1,x}$ where

$$Y_{2,x} = \varphi(Y_{1,x}) + Y_{d,x}.$$

As before, we will assume that $Y_{d,x}$ is independent of $Y_{1,x}$. We will also assume that $E[Y_{d,x}] = 0$.

The idea of locally linear polynomial regression is to approximate $E[Y_{2,x}|Y_{1,x} = y] = \varphi(y)$ with a linear function by using weighted data in which we give more importance to the observations made near y.

Since $Y_{1,x}$ and $Y_{d,x}$ are independent, conditioning by $Y_{1,x}$ does not change the distribution of $Y_{d,x}$. We might as well consider that the $Y_{1,x}$ are fixed to simplify the notation. Instead of conditioning we will estimate $E[Y_{2,x}] = \varphi(y_{1,x})$. In the next section we will deal with the fact that $Y_{1,x}$ is unknown.

We measure the importance of a data point by determining how close to $y_{1,x}$ it was observed. This distance will be defined by a kernel K that has some particular properties that may change depending on what type of behavior we are looking for. Two popular choices for K are

$$\begin{aligned} K_{Gaussian}(y) &:= \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \\ K_{Epanechnikov}(y) &:= 0.75(1-y^2)1_{\{-1\leq y\leq 1\}}(y). \end{aligned}$$

This distance between where we want to predict $y_{1,x}$ and where we observed a data point $y_{1,x'}$ will be measured by

$$K_h(y_{1,x'} - y_{1,x}) := \frac{1}{h} K\left(\frac{y_{1,x'} - y_{1,x}}{h}\right).$$

We choose a K_h that attains its maximum at 0 and then decays, symmetrically, giving higher weight to data that is observed near the point of prediction $y_{1,x}$.

With all of this in mind, for a given h, we solve

$$\underset{a_0,a_1}{\text{minimize}} \quad \sum_{x_i \in \mathcal{X}_2} \left(a_0 + a_1 \left(y_{1,x_i} - y_{1,x}\right) - y_{2,x_i}\right)^2 K_h \left(y_{1,x_i} - y_{1,x}\right) \tag{3.4}$$

and estimate $\varphi(y_{1,x})$ by comparing terms of the linear function on the objective above with the terms of the first order approximation of $\varphi(y_{1,x})$ around $y_{1,x}$. This is, the estimates for a_0 and a_1 are our approximations of $\varphi(y_{1,x})$ and $\varphi'(y_{1,x})$.

Problem (3.4) has an explicit solution that in matrix form can be written as

$$\begin{bmatrix} \widehat{a}_0 \\ \widehat{a}_1 \end{bmatrix} := \left(\begin{bmatrix} \mathbf{1}(n_2) & y_1(n_2 - x) \end{bmatrix}^T K_h(n_2; x) \begin{bmatrix} \mathbf{1}(n_2) & y_1(n_2 - x) \end{bmatrix} \right)^{-1} \\ \begin{bmatrix} \mathbf{1}(n_2) & y_1(n_2 - x) \end{bmatrix}^T K_h(n_2; x) y_{2,n_2}. \tag{3.5}$$

In the expression above, $K_h(n_2; x)$ is the diagonal matrix with entries $K_h\left(y_{1,x_i} - y_{1,x}\right)$ for $x_i \in \mathcal{X}_2$. The vector $y_1(n_2 - x)$ contains all the $(y_{1,x_i} - y_{1,x})$ differences. With this notation the locally linear estimate for $\varphi(y_{1,x})$ is

$$\widehat{\varphi}(y_{1,x}) := \widehat{a}_0.$$

The estimate depends on $y_{1,x}$. If we want to compute $\widehat{\varphi}$ at a different $y_{1,x'}$ we will have to compute a new $\widehat{a}_0$. This might seem costly but the matrix we have to invert each time we need to make new predictions is always 2×2.

This is how we are going to estimate the relationship between 2 successive fidelity levels.

Multi-fidelity regression with an estimated relationship

Now we go back to our prediction problem. This is, given an input x, we want to predict $Y_{2,x}$ using $(\mathcal{X}_1, y_{1,n_1})$ and $(\mathcal{X}_2, y_{2,n_2})$.

The model we use for the data is an extension of the multi-fidelity regression model based on Gaussian processes. We assumed that

$$Y_{2,x} = \varphi(Y_{1,x}) + Y_{d,x}$$

where φ is unknown function. Now we will assume that

$$\begin{aligned} Y_{1,x} &\sim GP(\mu_1,\ \sigma_1^2\gamma_1(x, x'; \theta_1)) \\ Y_{d,x} &\sim GP(0,\ \sigma_d^2\gamma_d(x, x'; \theta_d)) \end{aligned}$$

are independent Gaussian processes.

We still have the same independence structure as in the last chapter. So, like before, we can write

$$f_{Y_{1,n_1},Y_{n_2}}(y_{1,n_1}, y_{n_2}) = f_{Y_{n_1}}(y_{n_1}) f_{Y_d(n_2)}(y_d(n_2))$$

where $y_d(n_2) := y_{2,n_2} - \varphi(y_1(n_2))$. Using this result we can derive the conditional mean

$$E[Y_{2,x}|Y_{n_1}, Y_{n_2}] = E[\varphi(Y_{1,x})|Y_{1,n_1}, Y_{2,n_2}] + E[Y_{d,x}|Y_d(n_2)]. \tag{3.6}$$

We are going to estimate $\varphi(Y_{1,x})$ and $\varphi(y_{2,n_2})$ to compute these formulas.

In the last section we built an estimator for $\varphi(y) = E[Y_{2,x}|Y_{1,x} = y]$ by solving a weighted least squares problem, (3.4). The solution, written in equation (3.5), is a function of $Y_1(n_2)$ and Y_{2,n_2}. We will estimate $\varphi(Y_{1,x})$ with $\widehat{\varphi}(\widehat{y}_{1,x})$ where

$$\begin{aligned} \widehat{y}_{1,x} &= E[Y_{1,x}|Y_{1,n_1}] \\ \widehat{\varphi}(y) &= \widehat{a}_0. \end{aligned}$$

We use an approximation for $Y_{1,x}$ that we plug in our approximation for $E[Y_{2,x}|Y_{1,x} = y]$. The first term of equation (3.6) is approximated by

$$E[\widehat{\varphi}(\widehat{y}_{1,x})|Y_{1,n_1}, Y_{2,n_2}] = \widehat{\varphi}(\widehat{y}_{1,x}).$$

For the second term, we evaluate the conditional expectation at $\tilde{y}_d(n_2) := y_{2,n_2} - \widehat{\varphi}(y_1(n_2))$. Our prediction for $Y_{2,x}$ given the data is

$$\widehat{\varphi}(\widehat{y}_{1,x}) + E[Y_{d,x_0}|Y_d(n_2) = \tilde{y}_{d,x}]. \tag{3.7}$$

If we replace all these estimations on the conditional mean we have

$$\tilde{y}_{2,x} := \widehat{\varphi}(\widehat{y}_{1,x}) + \tilde{y}_{d,x}.$$

To summarize, in order to make a prediction at an input x, we use the following algorithm:

Algorithm 2 Multi-fidelity regression with estimated relationship.

1: **procedure** NPCK
2: Compute $\widehat{y}_{1,x} := E[Y_{1,x}|Y_{1,n_1} = y_{1,n_1}]$.
3: Compute $\widehat{\varphi}(\widehat{y}_{1,x})$ and $\tilde{y}_d(n_2) := y_{2,n_2} - \widehat{\varphi}(y_1(n_2))$.
4: Predict $y_{2,x}$ with $\widehat{\varphi}(\widehat{y}_{1,x}) + E[Y_{d,x}|Y_d(n_2) = \tilde{y}_d(n_2)]$.

The prediction of this algorithm is a mixture of Gaussian process linear regression and nonparametric estimation.

A first remark is that the prediction mean interpolates the data regardless of the estimated relationship. This is because if we take $x_j \in \mathcal{X}_2$, then

$$\begin{aligned}\widehat{\varphi}(\widehat{y}_{1,x_j}) + E[Y_{d,x_j}|Y_d(n_2) = \tilde{y}_{d,x}] &= \widehat{\varphi}(y_{1,x_j}) + y_{2,x_j} - \widehat{\varphi}(y_{1,x_j}) \\ &= y_{2,x_j}.\end{aligned}$$

A second remark is that we use $(\mathcal{X}_1, y_{1,n_1})$ to estimate the parameters of Y_1 and $(\mathcal{X}_2, \tilde{y}_d(n_2))$ to estimate those of Y_d.

Finally, this is not the usual model used in the theory of locally linear

polynomials. There are two main differences: the inputs and the measurement errors, modeled by $Y_{1,x}$ and $Y_{d,x}$, are correlated random variables.

There is some research for the case of correlated errors but the correlation structure we consider is different. For example in [40] by Jean Opsomer and in the PHD thesis of Xiao-Hu Liu [37] they assume that the correlation of the errors depends on Y_1. In our configuration their hypothesis would be written

$$Cov(Y_{d,x}, Y_{d,x'}|Y_{1,x}, Y_{1,x'}) = \rho(Y_{1,x} - Y_{1,x'}).$$

Instead we have that

$$\begin{aligned} Cov(Y_{d,x}, Y_{d,x'}|Y_{1,x}, Y_{1,x'}) &= Cov(Y_{d,x}, Y_{d,x'}) \\ &= \sigma_d^2 \gamma_d(x, x'; \theta_d). \end{aligned}$$

This is an original nonparametric regression problem motivated by its application to multi-fidelity regression.

3.1.2 Estimating the bandwidth parameter h

The bandwidth, h, is a tuning parameter that controls the bias and the variance of the estimator.

A very big h produces an estimate that takes into account all the data points. This estimate is equivalent to the best linear fit over the whole domain. So a big h, makes predictions with low variance that are probably far from the true. The estimate for a very small h will change a lot from one point to the next.

The bandwidth parameter is unknown and we have to fix it. To estimate it we minimize an approximation of the test error. There are two popular possibilities, using a penalty on the training error or cross validation.

For the penalty approach we approximate the test error on a set observed

on the same inputs as the training set. This is, the test set is $(y_1(n_2), Y'_{2,n_2})$ where Y_{2,n_2} and Y'_{2,n_2} are i.i.d. For ease of notation we assume that the inputs are fixed, otherwise we would write everything conditionally on $Y_1(n_2)$. The error on the test set is

$$\begin{aligned}\frac{1}{n_2}E\|Y'_{2,n_2}-\widehat{\varphi}(y_1(n_2))\|_2^2 &= \cdots \\ \cdots &= \frac{1}{n_2}E\|Y'_{2,n_2}-\varphi(y_1(n_2))\|_2^2 + \frac{1}{n_2}E\|\varphi(y_1(n_2))-\widehat{\varphi}(y_1(n_2))\|_2^2 \\ &\quad + \frac{2}{n_2}E(Y'_{2,n_2}-\varphi(y_1(n_2)))^T(\varphi(y_1(n_2))-\widehat{\varphi}(y_1(n_2)))\end{aligned}$$

The first term is σ_d^2 and the cross term is zero: the test and training samples are independent. We have that

$$\begin{aligned}\frac{1}{n_2}E\|Y'_{2,n_2}-\widehat{\varphi}(y_1(n_2))\|_2^2 &= \cdots \\ \cdots &= \sigma_d^2 + \frac{1}{n_2}E\|\varphi(y_1(n_2))-Y_{2,n_2}\|_2^2 + \frac{1}{n_2}E\|Y_{2,n_2}-\widehat{\varphi}(y_1(n_2))\|_2^2 \\ &\quad + \frac{2}{n_2}E(\varphi(y_1(n_2))-Y_{2,n_2})^T(Y_{2,n_2}-\widehat{\varphi}(y_1(n_2))) \\ &= 2\sigma_d^2 + \frac{1}{n_2}E\|Y_{2,n_2}-\widehat{\varphi}(y_1(n_2))\|_2^2 \\ &\quad + \frac{2}{n_2}E(\varphi(y_1(n_2))-Y_{2,n_2})^T(Y_{2,n_2}-\widehat{\varphi}(y_1(n_2))).\end{aligned}$$

We expand the cross term by distributing the right parenthesis to get two terms. Since $E(\varphi(y_1(n_2)) - Y_{2,n_2}) = EY_d(n_2) = 0$ we can add some additional constants to each one of the terms to have

$$\begin{aligned}\frac{2}{n_2}E(\varphi(y_1(n_2))-Y_{2,n_2})^T(E\widehat{\varphi}(y_1(n_2))-\widehat{\varphi}(y_1(n_2))) &= \frac{2}{n_2}tr\left(Cov(Y_{2,n_2},\widehat{\varphi}(y_1(n_2)))\right., \\ \frac{2}{n_2}E(\varphi(y_1(n_2))-Y_{2,n_2})^T(Y_{2,n_2}-\varphi(y_1(n_2))) &= -2\sigma_d^2.\end{aligned}$$

Finally, we have that the error on the test set can be written in function of the training error as

$$\frac{1}{n_2}E\|Y'_{2,n_2}-\widehat{\varphi}(y_1(n_2))\|_2^2 = \frac{1}{n_2}E\|Y_{2,n_2}-\widehat{\varphi}(y_1(n_2))\|_2^2 + \frac{2}{n_2}tr\left(Cov(Y_{2,n_2},\widehat{\varphi}(y_1(n_2)))\right..$$

We can write $\widehat{\varphi}(y_1(n_2)) = S(h)Y_{2,n_2}$, where $S(h)$ is a matrix that depends on h and $y_1(n_2)$. Then, $Cov(Y_{2,n_2}, \widehat{\varphi}(y_1(n_2))) = S(h)\Gamma_2(n_2)$. We select h by solving

$$\underset{h}{\text{minimize}} \quad \frac{1}{n_2}\|Y_{2,n_2} - \widehat{\varphi}(y_1(n_2); h)\|_2^2 + \frac{2}{n_2} tr\left(S(h)\Gamma_2(n_2)\right). \tag{3.8}$$

The objective in (3.8) is an unbiased estimator of the expected test error.

The second method is cross validation. As before we separate the data into V disjoint sets with indexes $F_1 \cup \cdots \cup F_V = \mathcal{X}_2$. We note $\widehat{\varphi}^{(-j)}(y_{1,x_j}; h)$ the local linear estimator built using the data indexed by $\mathcal{X}_2 \setminus F_j$ and compute its test error on F_j

$$CV_j(h) := \frac{1}{|F_j|}\sum_{k \in F_j}\left(Y_{2,x_k} - \widehat{\varphi}^{(-j)}(y_{1,x_j}; h)\right)^2.$$

The cross validation error is the average of the test errors

$$CV(h) := \frac{1}{V}\sum_{j=1}^{V} CV_j(h).$$

Because $\widehat{\varphi}(y_1(n_2)) = S(h)Y_{2,n_2}$, we are working with a linear smoother and there is a convenient formula for the case $V = n_2$ for which we do not need to compute $\widehat{\varphi}^{(-j)}(y_{1,x_j}; h)$:

$$CV_{n_2}(h) := \frac{1}{n_2}\sum_{k=1}^{n_2}\left(\frac{Y_{2,x_k} - \widehat{\varphi}(y_{1,x_k}; h)}{1 - S(h)_{(k,k)}}\right)^2.$$

All of these approximations make sense if we work with independent data. In [37] Liu proposes the corrected formula

$$CV^c_{n_2}(h) := \frac{1}{n_2}\sum_{k=1}^{n_2}\left(\frac{Y_{2,x_k} - \widehat{\varphi}(y_{1,x_k}; h)}{1 - (S(h)\Gamma_2(n_2))_{(k,k)}}\right)^2$$

in which we take into account the correlation of the data. A second option

is to select h by solving (3.9) below.

$$\underset{h}{\text{minimize}} \quad CV^c_{n_2}(h). \tag{3.9}$$

In practice we are going to look for the optimal h on a grid $[1/n_2, C]$ where C is proportional to the range of the data.

3.1.3 Prediction error

To build our prediction at a new input x we began by showing that the likelihood of the data can be decomposed. Using this fact we concluded that

$$E[Y_{2,x}|Y_{1,n_1}, Y_{2,n_2}] = E[\varphi(Y_{1,x})|Y_{1,n_1}, Y_{2,n_2}] + E[Y_{d,x}|Y_d(n_2)]$$

and in Algorithm 2 we estimated the different parts of this conditional expectation. The final prediction was

$$\widehat{\varphi}(\widehat{y}_{1,x}) + E[Y_{d,x_0}|Y_d(n_2) = \tilde{y}_{d,x}].$$

For the conditional variance we have a similar formula

$$Var(Y_{2,x}|Y_{1,n_1}, Y_{2,n_2}) = Var(\varphi(Y_{1,x})|Y_{1,n_1}) + Var(Y_{d,x}|Y_d(n_2)).$$

From this equation we can get the difference process prediction error by using the fact that it is Gaussian. The first term is more complicated because we do not know the distribution of $\varphi(Y_{1,x})$.

For the first term we use the Taylor approximation of $\varphi(Y_{1,x})$ around the conditional mean $\widehat{y}_{1,x}$ and we take the variance, conditionally on Y_{1,n_1}, to get

$$Var(\varphi(Y_{1,x})|Y_{1,n_1}) \approx (\varphi'(\widehat{y}_{1,x}))^2 \, Var(Y_{1,x}|Y_{1,n_1})$$

We have two terms: the first depends on an unknown derivative and for the

second we have an analytic expression. Fortunately, we have an estimate for this derivative, it is $\widehat{a}_1$ given by equation (3.5). Our prediction error will be

$$(\widehat{a}_1)^2 \, Var(Y_{1,x}|Y_{1,n_1}) + Var(Y_{d,x}|Y_d(n_2)). \tag{3.10}$$

This prediction error is 0 at the inputs of the data.

3.2 Illustrative examples

Simulated data

The first test consist in simulating data from two independent Gaussian processes with Matern 5/2 correlation functions

$$\begin{aligned} Y_{1,x} &\sim GP(0,\ \sigma_1^2 \gamma_1(x, x'; \theta_1)) \\ Y_{d,x} &\sim GP(0,\ \sigma_d^2 \gamma_d(x, x'; \theta_d)) \end{aligned}$$

and building the objective function by using three different relationships

$$\begin{aligned} Lin(y) &:= 2.5y \\ Sig(y) &:= 3/2(1 + e^{25y-2}) - 3/4 \\ Sin(y) &:= \sin(6y). \end{aligned}$$

shown in the plots in figure (3.3). The relationships are increasingly non-linear.

We compare our method that we note $npCK$ with three alternatives: a 1 data set model K and two multi-fidelity Gaussian process models CK and $CK(x)$. They are implemented in the R packages DiceKriging and MuFiCokriging.

The main difference between all these models is the form of the rela-

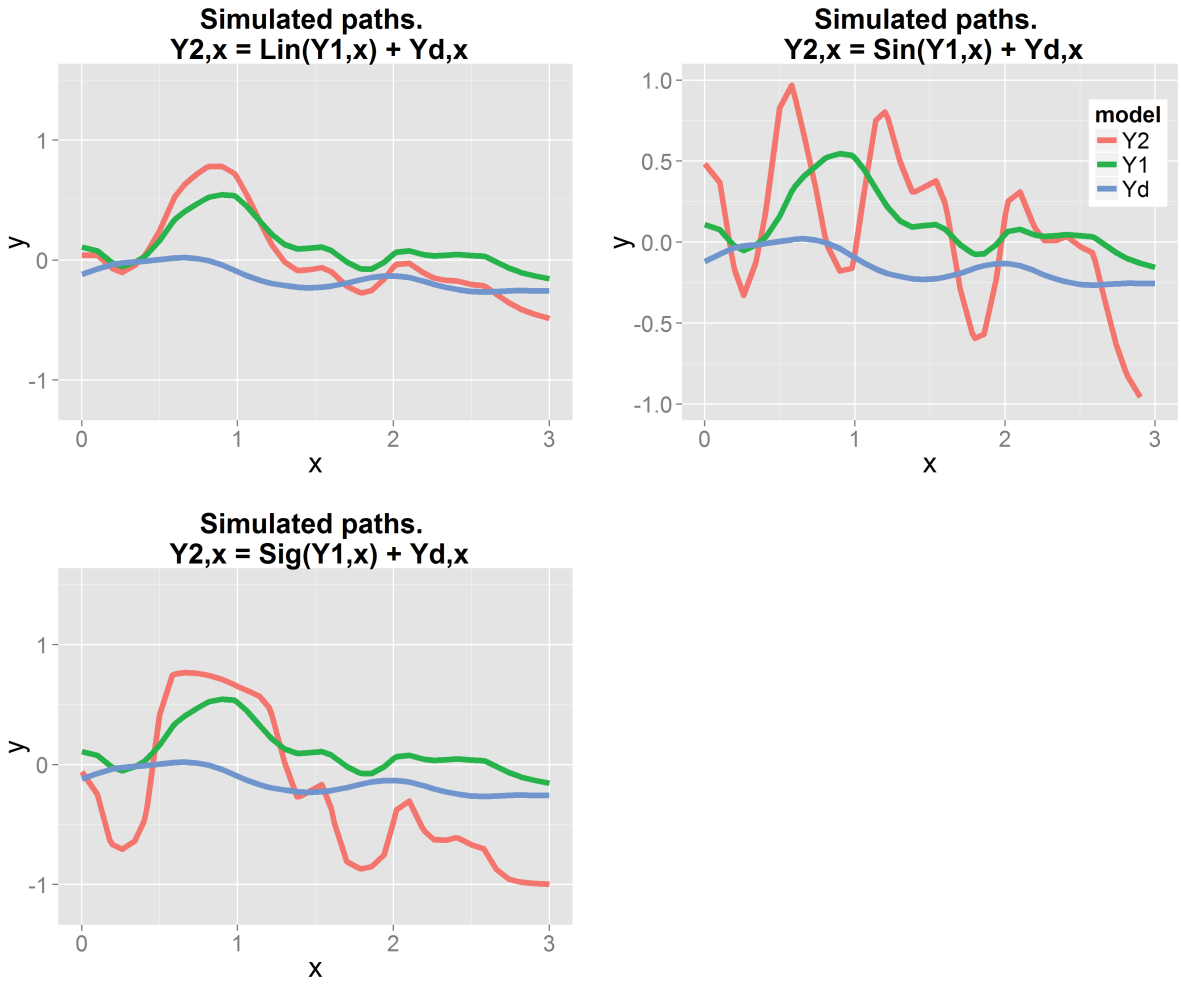


Figure 3.2: Simulated paths of the Gaussian processes $Y_{1,x} \sim GP\left(0, 0.4^2\gamma_1(x, x'; 0.5)\right)$ and $Y_d \sim GP\left(0, 0.2^2\gamma_1(x, x'; 0.8)\right)$. The target process $Y_{2,x}$ is built using one of the three relationships in each panel.

tionship assumed. The models for the target process Y_2 are

$$\begin{aligned}
(K): &\ Y_{2,x}, \\
(CK): &\ Y_{2,x} = \beta Y_{1,x} + Y_{d,x}, \\
(CK(x)): &\ Y_{2,x} = (\beta_0 + \beta_1 x) Y_{1,x} + Y_{d,x}, \\
(npCK): &\ Y_{2,x} = \varphi(Y_{1,x}) + Y_{d,x}
\end{aligned}$$

We simulate 100 paths and we sample them to get two training sets with $n_1 = 8$ and $n_2 = 16$ points and a test set with $n_{test} = 36$ points sampled on a regular grid. Some sample paths are shown in figure 3.2.

The boxplots of the $RMSE$ over the test set are shown on figure 3.4. For

all the relationships the RMSE for $npCK$ is considerably smaller although we are in a difficult configuration for multi-fidelity: the test error for CK and $CK(x)$ are bigger than that for K.

By estimating the relationship between two successive levels we can adapt to a wide range of data sets without making a lot of additional assumptions on the model.

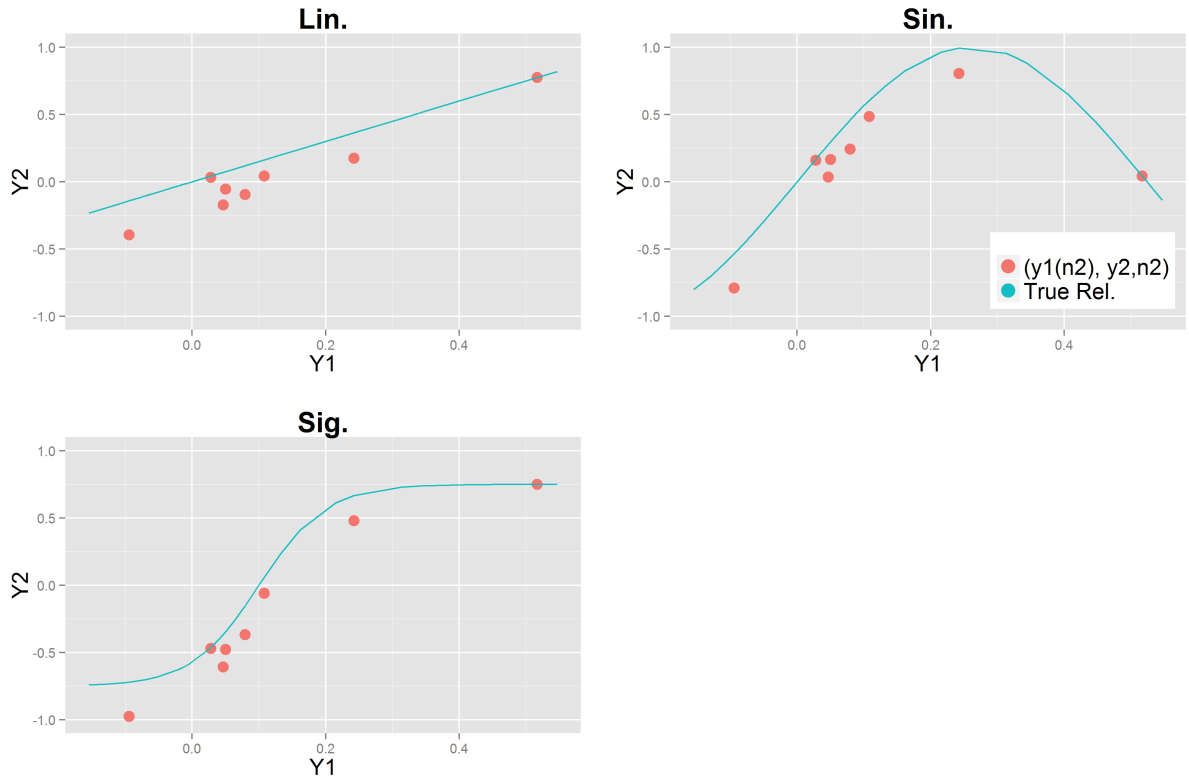


Figure 3.3: Relationship between the simulated data $y_1(n_2)$ and y_{2,n_2} sampled from the paths on figure 3.2. The pink points represent the training data over $\mathcal{X}_2$ and the blue lines are the real relationship. From the top left they are $Lin(y) := 2.5y$, $Sig(y) := 3/2(1 + e^{25y-2}) - 3/4$ and $Sin(y) := \sin(6y)$.

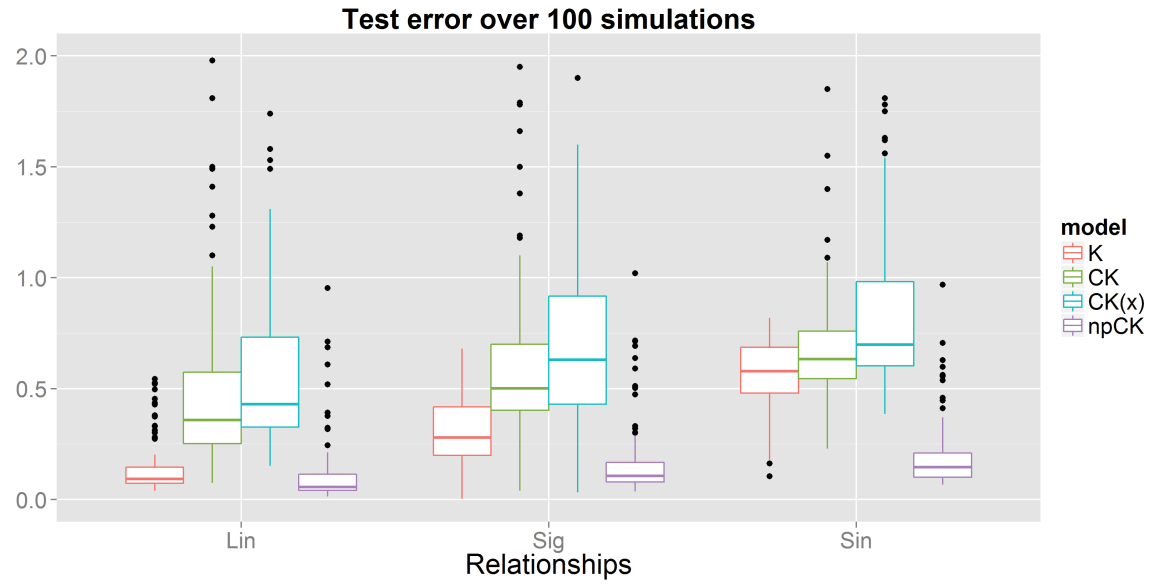


Figure 3.4: RMSE on x_{test} for the models studied. Each panel represents one of the three relationships used to build Y_2.

Numerical solution of the Darcy equation

We go back to the example at the beginning of the chapter. For a given mesh size N and a given combination of parameters (K, n^*), we solve the differential equation (3.3) numerically over the whole domain and consider its maximum value $p(K, n^*; N)$ as the output.

The target data is $p(K, n^*; N = 40)$ and we are interested on the relationship shown on the first panel figure 3.1.

Figure 3.5 shows the contour plots of target function $N = 40$ and its approximation $N = 5$. They share a similar structure, the picture is divided into 4 quadrants and the pattern repeats from left to right. They have some important differences too. Most notably, the values on the top of the plots are almost 0 for the target function and very high for the approximation and there is a valley on the bottom part of the target function that is not present on the other.

We sample the two functions to get two training sets that have $n_1 = 34$ and $n_2 = 17$ points. They are shown as dots in figure 3.5. The test set is

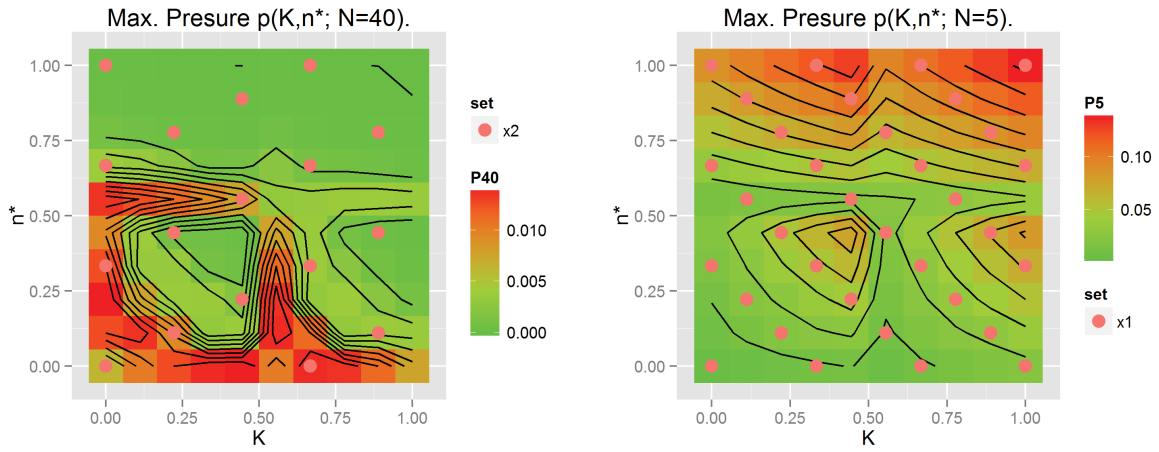


Figure 3.5: Contour plots of the maximum pressure for $N = 40$ and $N = 5$. The dots represent the locations of the training set data.

fine grid on the whole domain.

We use these training sets to build the 4 models we introduced in the last section. This is K, CK, $CK(x)$ and our method $npCK$.

The predictions are shown on figure 3.6. The first model K shows what can be achieved by using $(\mathcal{X}_{n_2}, y_{2,n_2})$ only. We have some idea of the structure but the valley on the first quadrant is missing.

The models on the left column of figure 3.6, CK and $CK(x)$, assume a parametric form for the relationship between the two outputs. The first model uses a constant and the second an affine function of the inputs (K, n^*). The prediction seems to be mislead by the approximate data: the height of the prediction on the top half coincides with that of $p(K, n^*; N = 5)$. The resulting picture is far away from the true function. In this case it seems preferable to use model K.

The last panel shows the $npCK$ prediction. We use the estimated relationship in figure 3.7 to incorporate the approximate data.

The top half of the last panel corresponds to the last part of the functions on 3.7. Because we predicted that those high values of $p(K, n^*; N = 5)$ correspond to a very low pressure $p(K, n^*; N = 40)$, we make an accurate

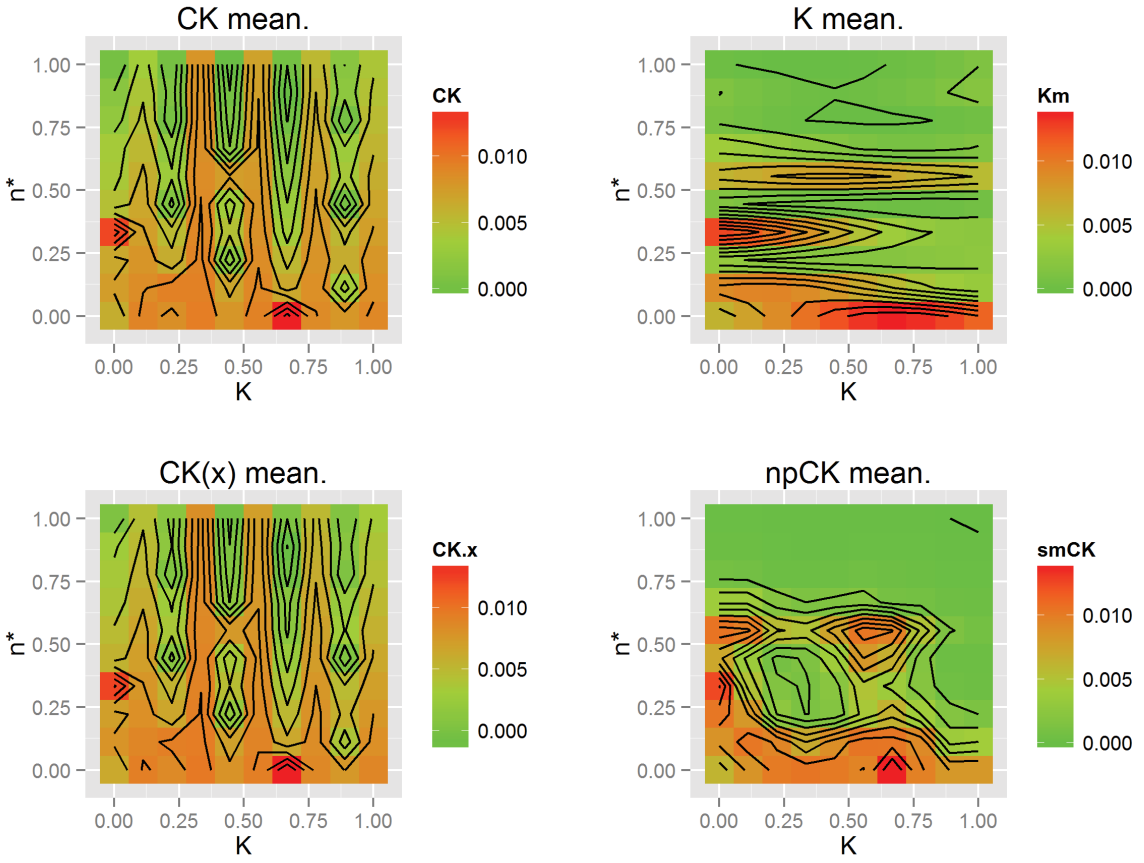


Figure 3.6: Contour plots of the predictions made by CK, $CK(x)$, K and $npCK$ on the test set, a fine grid of points on the whole domain.

prediction.

The predicted relationship also helps the model to discover the valley on the bottom left part.

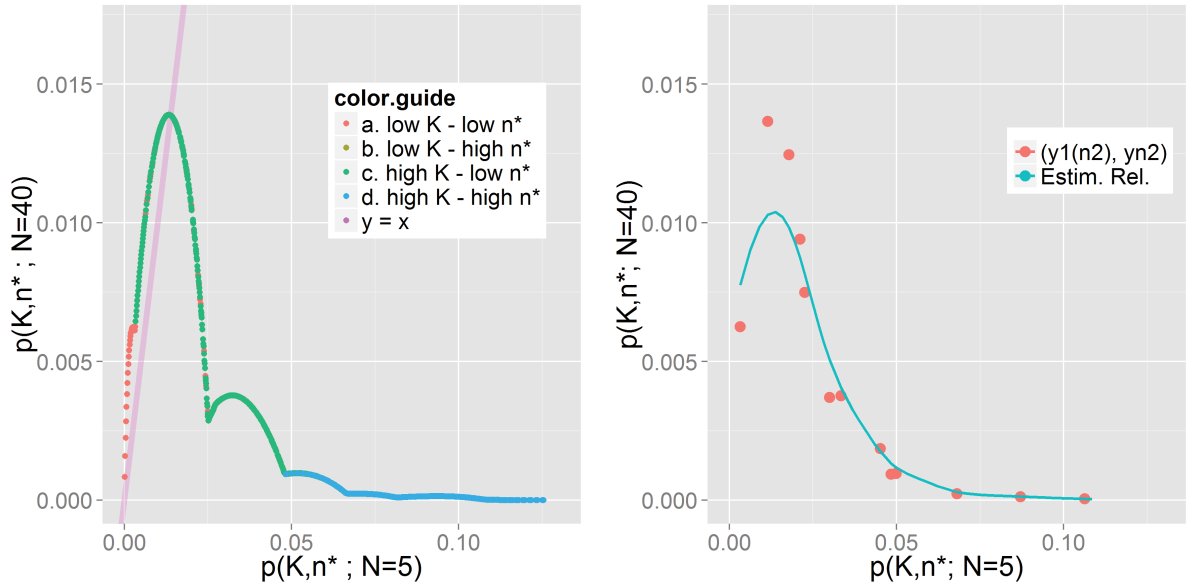


Figure 3.7: The left pannel is the true relationship between the data sets. Each color corresponds to a quadrant on the domain of (K, n^*). The right panel shows the relationship estimated using $(y_1(n_2), y_{2,n_2})$.

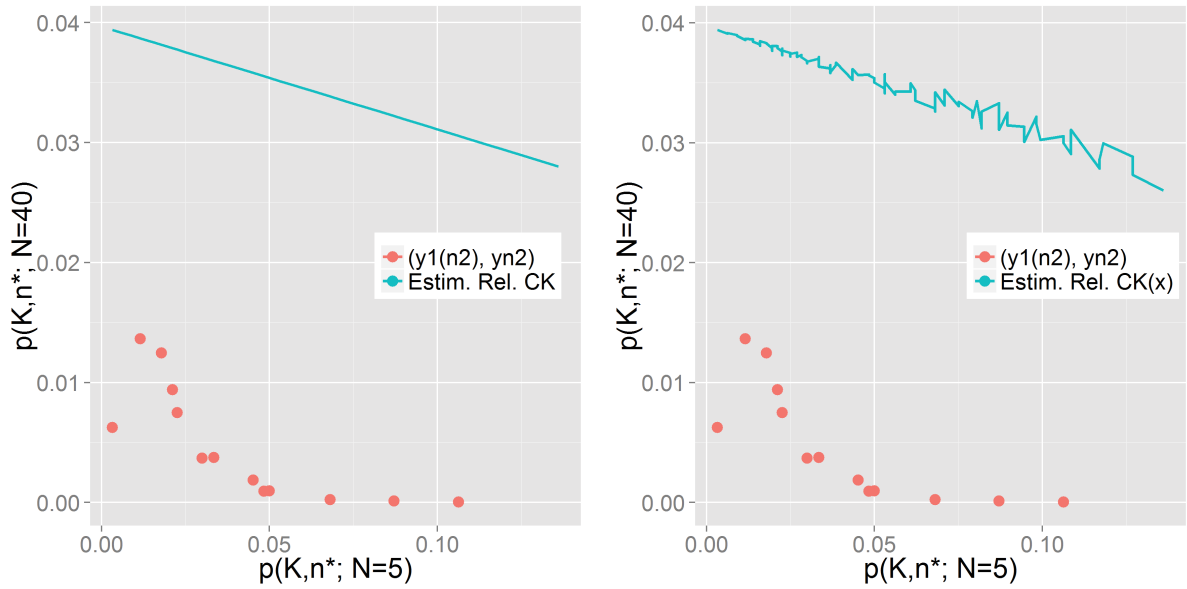


Figure 3.8: The left pannel shows the estiamted relatiosnhip for CK, $\beta y_{1,x}$. The right panel shows the relationship estimated for $CK(x)$, $(\beta_0 + \beta_1 x) y_{1,x}$. We add the estimated mean of the difference process to the relationship. We see that the slope is correct but the estimated mean of the difference is off.

3.3 Conclusion

Unknown relationship. In this chapter we extend the multi-fidelity model based on Gaussian processes with parametric relationship. We assume the same dependence structure and that we are given an order of fidelity. The difference is that the relationship between two successive levels, φ, is now unknown. We use locally linear polynomials to estimate φ. When the input sets of the data are nested we can build a predictor with its prediction error sequentially.

By estimating the relationship we are able to incorporate data that would be useless otherwise and we show that in two numerical examples. The estimated relationship adds useful information about the data.

New nonparametric regression problems. The hypothesis we use in our model are very different from those of the usual setting for nonparametric regression. We have not fully explored the properties of this locally linear estimator but the multi-fidelity model motivates a new class of nonparametric estimators that are worth studying.

The model proposed in this chapter simplifies the Bayesian structure of the parametric multi-fidelity model. Its main drawback is that the properties of the locally linear estimator are not fully described. The main difficulty is the fact that we are working with Gaussian processes with a complex covariance structure. In the next chapter we will apply this model to a case study of a diphasic air-water flow in rectangular domain.

Chapter 4

Case Study

In this chapter we present a case study related to an air-water flow on a rectangular pipe that approximates a part of the emergency core cooling system of a nuclear power plant. This flow is modeled by a numerical computer code that depends on a series of parameters. The users of this numerical simulator are uncertain about the precise values of the parameters and they would like to know what is their influence on some particular outputs. Because of time constraints we cannot use the simulator directly. We replace it with a Gaussian process based model.

The objectives of this chapter are first to compare the different models described in the previous chapter and second to measure the advantages of using multi-fidelity models when performing a Sensitivity Analysis on the parameters.

We start the chapter by explaining the motivation behind this computer simulation code. Then, we describe the problem formally by defining the parameters and the outputs we are interested in. We perform a first exploratory analysis to determine the influence of the parameters by using a fractional factorial design. Then, we compare the prediction accuracy of two multi-fidelity models to finally use these models to perform a Sensitivity Analysis on the parameters.

4.1 Motivation

The case study motivation is to have a better understanding of the lifetime of a nuclear power plant. The aging of a nuclear power plant is defined by the International Atomic Energy Agency (IAEA) as a continuous time-dependent loss of quality of materials, caused by the operating conditions.[1]

Aging processes are difficult to detect because changes happen in a microscopic level in the materials inside the power plant.

One of the possible consequences of this changes is the break of a pipe caused by a leak in the Emergency Core Cooling system of the power plant. The cold water flowing from the leak mixes with the very hot water contained in the cold leg. This mixture flows through the cold leg to what is called the downcomer causing high thermal gradients in the surrounding materials. This change in temperature could lead to a thermal shock on the reactor pressure vessel (RPV) wall. Such a scenario may lead to extreme thermal gradients in the structural components and consequently to very high stresses. Therefore, the loads upon the RPV must reliably be assessed. Precise knowledge of flow in the cold leg and downcomer is necessary to predict these thermal gradients in the structural components of the wall. A schematic representation of the accident is described by figure 4.1.

We study a two-phase, air-water, stratified flow that is a simplified representation of the flow in the cold leg. To validate this approximation it is essential to model the heat and mass transfer on the free surface and thus the flow precisely. The outputs we consider are some of the defining quantities of this flow on the interphase. They are the mean speed of the air and the water and the water level.

The model is implemented in the NEPTUNE CFD platform. The test case used is called Air-Water STratified (AWST) where stratified means

[1] Safety Aspects of Nuclear Power Plant Aging; TECDOC-540, International Atomic Energy Agency, Vienna, 1990

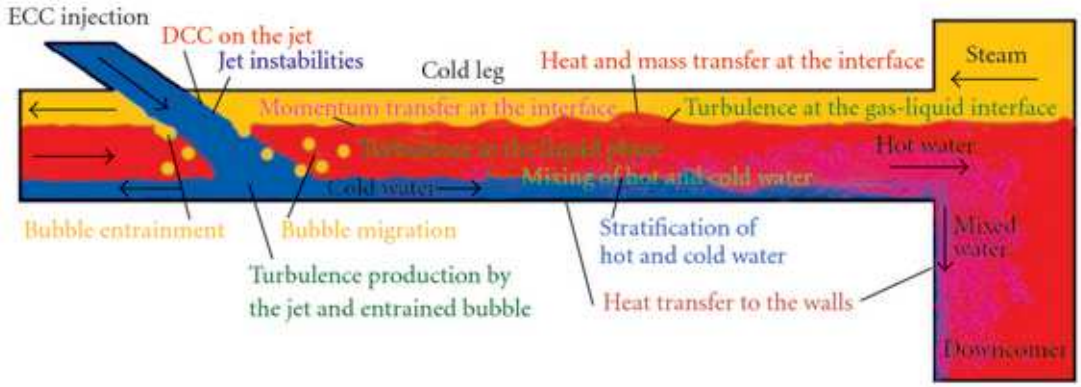


Figure 4.1: Water injection to the cold leg.

that the two fluids mix only near the interphase. It is based on the Fabre et al. experiment described in [21].

More specifically, we focus on the influence of some parameters on the outputs produced by the AWST test case. Some of these parameters come from the modeling of the mono-phasic flows; some are used to define the free surface and some have a numerical nature. We are uncertain on the value of these parameters and we only dispose of some reference values. For example, we are uncertain about the value of the parameter Re_s that defines the limit between a rough and a smooth interphase. We dispose of two reference values taken from an experiment in which a fluid is modeled as spheres or sand grains and this parameter could take any other value in between.

Our objective is to perform a sensitivity analysis on the parameters based on the Sobol indexes. In order to do this we need to run the AWST test several times. Since it takes around 10 minutes to produce a single output, performing the sensitivity analysis using the NEPTUNE CFD simulations directly is unfeasible. We would like to build a reliable and fast meta-model on a few runs to perform the sensitivity analysis on it instead. Even though we are studying an approximate model of the real flow, we can already see the importance of a metamodel.

A particularity of this case study is the fact that we dispose of different meshes with which we can compute the outputs of the AWST test. The time it takes to generate an output and its quality are dependent on the number of grid-cells used to build the mesh on the domain. We study a regression model that incorporates the outputs computed using different meshes to enhance the prediction quality.

4.2 Case Description

The physical model is based on the experiment by Fabre et al. [21] where they consider a channel 12 m long, with a rectangular cross section of 0.2 m wide and 0.1 m high. The domain is shown in figure 4.2.

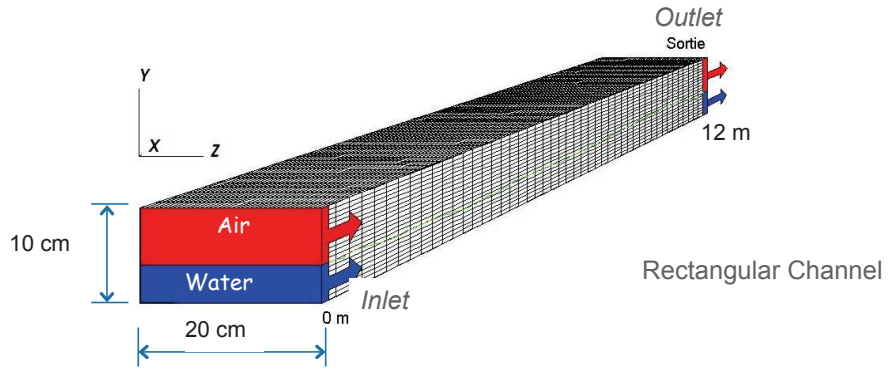


Figure 4.2: The domain in which we solve the flow equations. It has a 0.1 percent slope.

The channel is supplied with air and water coming from one of the sides. There is a 0.1 percent bottom slope and the superficial water velocity at the inlet is 0.15 m/s which corresponds to a flow rate of 3 l/s.

There are two flow regimes. They depend on the air flow rate on the inlet. When the flow rate is 45.4 l/s the interface is smooth and when it is 75.4 l/s it becomes wavy. We consider the wavy regime.

The measurements are made at 9.1 m from the inlet. Since we are interested in the mass and heat transfer on the free surface we extract the simulations computed at the air-water interphase shown on figure 4.3. We

consider the mean over 30 seconds of several quantities: the gas and water speeds G and W and the water level L as done in [12].

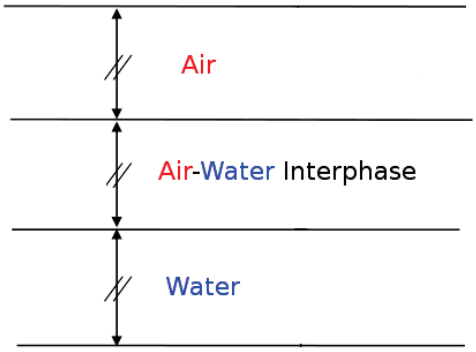


Figure 4.3: The cross section of the domain is divided into three different parts, air, water and a middle section where the two fluids meet. We focus on the middle interphase.

Finally, the turbulence in the air and water is a $k - epsilon$ model [4].

We study the influence of 9 parameters, defined in [11], on the outputs. They are

- 3 constants related to the turbulence model of the mono-phasic flow: $CE1$, $CE2$, $CEMU$;
- 2 Reynolds coefficients: $Re_s = su^*/\nu$ and $Re_r = ru^*/\nu$, that define the limit between two types of flow regimes - rough and smooth. Where s and r are the mean height of the waves in the surface or roughness; u^* is a dimensionless speed and ν is the viscosity;
- a correction factor Δ_μ that represents the difference between the liquid and interphase velocities, defined by $(1+\Delta_\mu)u'_{32}$ where u'_{32} is the limit velocity between different flow regimes;
- a coefficient β_1 defining roughness due to the gas friction as $r = \beta_1 \frac{u^*_G}{g}$. Where u^*_G is the gas speed, g is gravity and r the roughness;
- a mono-pahsic coefficient used to compute the friction on the walls $Cdupdyp$;

- a coefficient q that acts a the limit between two regimes defined vaguely by the turbulence velocity of the blobs.

We are uncertain about the real value of the parameters. We assume that these parameters can vary over closed intervals uniformly. Most of the bounds of the intervals can be found in [11]. Those of $CE1$, $CE2$ and $CEMU$ were fixed by an expert research engineer in thermohydraulics simulation at the CEA.

- The intervals $CE1 \in [1.3, 1.6]$, $CE2 \in [1.7, 2.2]$ and $CEMU \in [0.07, 0.1]$ are defined by an expert;
- $Re_s \in [2.25, 15]$ and $Re_r \in [55, 90]$ are taken from a reference in [11]. The limits depend on two different shapes of the roughness - sand grains or spheres - and they are recommended values used in practice. We do not know what model to use or if we should use something in between;
- $\Delta_\mu \in [0.1, 0.9]$ takes values between almost 0 and almost 1. There is always a difference between the liquid and interphase velocities but we are uncertain of how big it is;
- $\beta_1 \in [0.05, 0.1]$ where the lower limit comes from oceanography literature and the upper bound from fully developed models of steady state stratified flows in pipes. We do not know which model is closer to our case;
- $Cdupdyp \in [0.7, 1.3]$ is fixed by an expert;
- $q \in [\sqrt{1/3}, 1]$ the uncertainty comes from trying to relate the turbulence velocity of the blobs to a velocity produced by the turbulence model. This parameter q is the key unknown quantity. The limits can be found in [11].

The equations governing the flow are solved by using the finite volume method. For a given set of parameters we compute the finite volumes over

3 different domain discretizations. We write the number of grid-cells in the (x, y, z) directions of figure 4.2 as $X \times Y \times Z$. Meaning X cells in the x directions Y in the y and Z in the z. We set $Z = 1$ because the results correspond to the physical manipulation of [21] and it is faster than a full $3D$ model.

The first mesh has $50 \times 20 \times 1$ cells; the second $50 \times 40 \times 1$ and the third $100 \times 40 \times 1$. From now on we use the subscripts 1, 2 and 3 to distinguish the outputs computed using each of the 3 discretizations listed above. This is our fidelity parameter and mesh 3 is our target output. The other two are considered as approximations.

The average time to produce an output on a high performance computer using a single combination of parameters is

- 2 minutes using mesh 1, $(50 \times 20 \times 1)$;
- 3 minutes using mesh 2, $(50 \times 40 \times 1)$;
- 10 minutes using mesh 3, $(100 \times 40 \times 1)$.

Because of the time it takes to produce an output, it is convenient to make a first exploratory study before building the multi-fidelity model on all the 9 parameters. The objective of this approach is to reduce the number parameters.

4.3 First approach: fractional factorial design

We use the letter x to denote the input vector of parameters. The quantities of interest y are the target solutions computed using the third mesh $(100 \times 40 \times 1)$.

We consider the interactions of up to 2 parameters under the hypothesis

that the output is a combination of the parameters written as:

$$y = \sum_{k=0}^{9} \beta_k x_k + \sum_{l<k} \beta_{(l,k)} x_l x_k + \epsilon, \quad \beta_k, \beta_{(l,k)} \in \boldsymbol{R}. \tag{4.1}$$

where $\epsilon \sim N(0, \sigma^2)$.

On the first experiment to determine the influence of the parameters, we sample the simulator over a 128 points fractional factorial design with resolution V. This design is a subset of all the possible inputs in equation 4.1 where $(x_1, ..., x_9)$ take their values at the vertexes of the hyper cube of all possible parameters and the interactions of order two are built computing their corresponding products. Resolution V means that some order two interactions are confounded with order three interactions. They can be computed from $x_8 = x_2x_3x_4$ and $x_9 = x_1x_2x_5x_6$.

We estimate the parameters on equation 4.1 and select the inputs according to t-statistic test like the one on the last column of table 4.1.

The results of fitting the model for the gas speed G_3 are shown on table 4.1.

	Estimate	Std. Error	t value	Pr(> |t|)
(Intercept)	2.720757	0.017100	159.106	< 2e-16 ***
Re_r	-0.034102	0.012927	-2.638	0.009500 **
β_1	0.004958	0.015832	0.313	0.754736
q	-0.034102	0.012927	-2.638	0.009500 **
$CE1$	-0.664057	0.015832	-41.945	< 2e-16 ***
$CE2$	0.025754	0.015832	1.627	0.106560
CMU	0.056227	0.015832	3.552	0.000558 ***
$Cdupdyp$	-0.038907	0.015832	-2.458	0.015496 *
$Re_r : q$	0.068205	0.018281	3.731	0.000299 ***
$\beta_1 : CMU$	-0.077884	0.018281	-4.260	4.22e-05 ***
$\beta_1 : Cdupdyp$	0.045358	0.018281	2.481	0.014554 *
$CE1 : CE2$	0.667875	0.018281	36.534	< 2e-16 ***
$CE1 : CMU$	0.050561	0.018281	2.766	0.006626 **
$CE2 : Cdupdyp$	0.091212	0.018281	4.989	2.19e-06 ***

Table 4.1: Parameters estimated using the fractional factorial design for the gas speed G_3.

The conclusion we draw from table 4.1 above is that all the parameters have an important linear influence except for one of the Reynolds coefficients, Re_s and the turbulent speed of the liquid Δ_μ. Their corresponding coefficients are not significant. They do not even appear through interactions.

We have similar results for the other outputs. For the lower fidelity levels, we have that less parameters are influential but neither Re_s or Δ_μ appear as in the results for G_2 shown on table 4.2.

	Estimate	Std. Error	t value	Pr(> |t|)
(Intercept)	2.69069	0.02150	125.126	< 2e-16 ***
β_1	-0.05937	0.02483	-2.391	0.018373 *
$CE1$	-0.70233	0.02027	-34.642	< 2e-16 ***
$CE2$	0.04705	0.02027	2.321	0.022000 *
CMU	0.15402	0.02027	7.597	7.61e-12 ***
$Cdupdyp$	-0.08092	0.02027	-3.991	0.000114 ***
$\beta_1 : CMU$	-0.13595	0.02867	-4.742	5.94e-06 ***
$\beta_1 : Cdupdyp$	0.16902	0.02867	5.895	3.58e-08 ***
$CE1 : CE2$	0.74610	0.02867	26.022	< 2e-16 ***

Table 4.2: Parameters estimated using the fractional factorial design for the gas speed G_2.

From this preliminary study we can see that degrading the mesh affects the influence of the parameters on the output. Also, it seems that less parameters are influential on the the liquid outputs. Table 4.3 below shows only 5 influential parameters on the water speed W_3.

In conclusion, we keep the remaining seven parameters to build the metamodels of the next section.

4.4 Metamodels tested

We reduced the number of parameters from 9 to 7 by using the sample over a fractional factorial design to fit the model of equation 4.1. To perform a Sensitivity Analysis we need to have a reliable metamodel adjusted to samples from all over the input domain.

Coefficients:	Estimate	Std. Error	t value	Pr($>\|t\|$)
(Intercept)	0.506247	0.002613	193.710	$<$ 2e-16 ***
β_1	-0.012157	0.002613	-4.652	8.79e-06 ***
$CE1$	0.096519	0.003018	31.984	$<$ 2e-16 ***
$CE2$	0.005795	0.002613	2.218	0.028535 *
CMU	-0.014267	0.003374	-4.229	4.71e-05 ***
$Cdupdyp$	-0.008027	0.002134	-3.762	0.000266 ***
$\beta_1 : CE1$	0.010575	0.003018	3.504	0.000651 ***
$\beta_1 : CMU$	0.006663	0.003018	2.208	0.029218 *
$CE1 : CE2$	-0.123280	0.003018	-40.852	$<$ 2e-16 ***
$CE1 : CMU$	-0.017550	0.003018	-5.815	5.44e-08 ***
$CE2 : CMU$	0.021158	0.003018	7.011	1.68e-10 ***
$CMU : Cdupdyp$	0.014695	0.003018	4.870	3.57e-06 ***

Table 4.3: Parameters estimated using the fractional factorial design for the water speed W_3.

In particular we are going to use the outputs produced by the different meshes to build multi-fidelity metamodels. In the next section we introduce a new algorithm to build space filling designs based on maximizing the entropy of the outputs.

4.4.1 The inputs

In this case study we dispose of three different discretizations of the rectangular pipe that we sample to build several multi-fidelity regression model based on Gaussian processes. We use nested input sets to have a sequential construction of the models as discussed in chapter 2.

We note X_1, X_2 and X_3 the input designs for the three meshes. They are nested as $X_3 \subset X_2 \subset X_1$. Besides the training inputs we build a test set to compare the different meta-models we study.

In the method we propose, to build X_1, X_2 and X_3 we start with a space filling design X_0. From X_0 we extract the subset X_1 that maximizes the entropy of the output. The output is supposed to be a Gaussian process. To maximize its entropy it is sufficient to define the covariance function.

Here we consider a spherical covariance

$$\gamma_S(x, x'; \theta) := \left(1 - \frac{3}{2}\frac{\|x - x'\|_1}{\theta} - \frac{1}{2}\left(\frac{\|x - x'\|_1}{\theta}\right)^3\right) 1_{\{\|x-x'\|_1 \leq \theta\}}(x, x').$$

as suggested in [24] section 2.3.8. In practice the covariance parameters are fixed to values that are big with respect to the domain. When working in $[0, 1]^D$, we use $\theta = \sqrt{D}$.

We extract X_2 from X_1 by using the same strategy and finally we extract X_3 from X_2. The test set is the difference between X_0 and X_1. Schematically we have

$$X_0 \to X_1 \to X_2 \to X_3$$
$$X_{test} := X_0 \setminus X_1.$$

For an initial design with good space filling properties, this strategy can produce well spread nested designs as can be seen on figure 4.4 below.

Selecting the maximum entropy designs can be a nice alternative to selecting the nested sets at random. We use the *NestedDesign* function implemented in the *MuFiCokriging* R package to produce nested designs selected at random. The sets are selected in the same order as in our method.

To test the two methods to build nested designs, we use the criteria described in the article by G. Damblin [13]. This is the mean and variance of the distances of the edges of the Minimum Spanning Tree and the discrepancy between the distribution of the points and a uniform distribution.

The Minimum Spanning Tree (MST) is a graph that joins all the points in the design such that the sum of the size of its edges is minimal. Intuitively, a good space filling design has a MST with big edges that have more or less the same size. This can be interpreted as a big mean and small variance on the size of the edges.

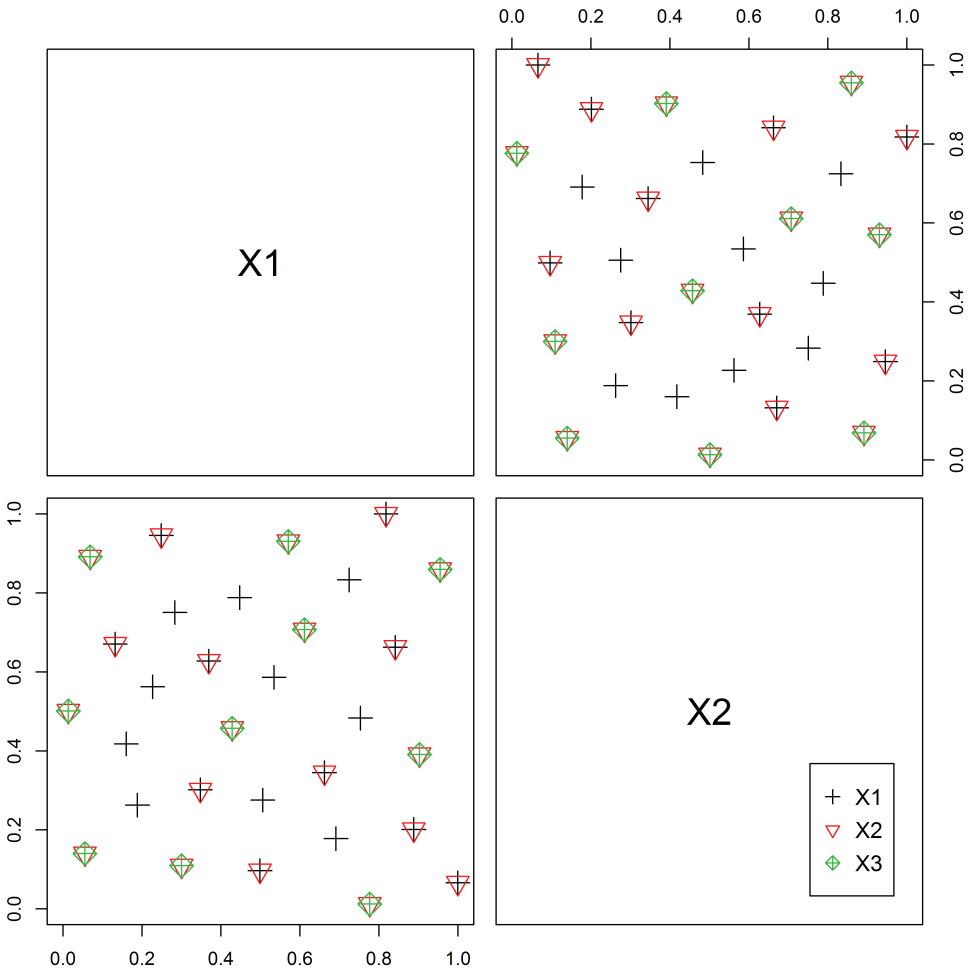


Figure 4.4: A two dimensional example of two *Dmax* nested designs built from the maximin LHS X_1.

A small discrepancy is a hint of uniformly distributed points. We use a C_2 discrepancy, the formula is given in [13].

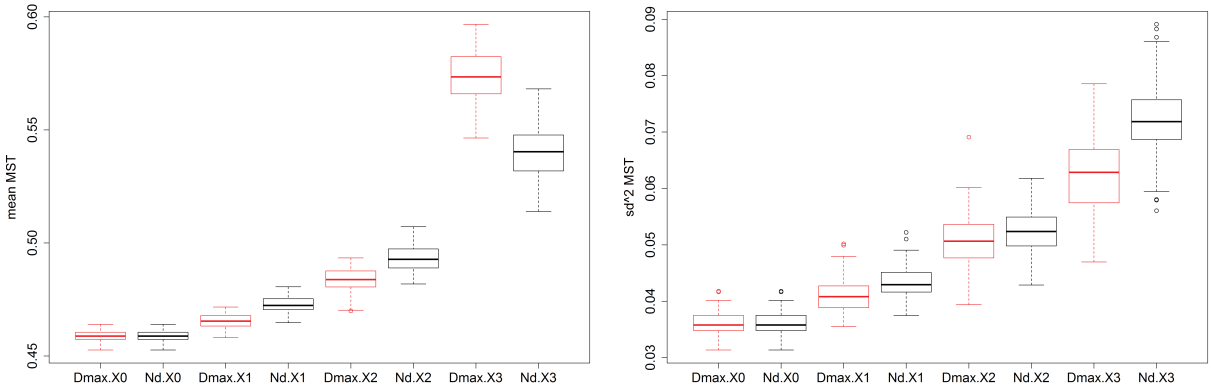
The mean and variance of the MST and the discrepancy functions are implemented in the *DiceDesign* R package.

The initial design X_0 is a space filling *lhs* with 300 points, X_1 has 210, and X_2 and X_3, 140 and 70 respectively. The test set X_{test} contains 90 points.

We draw 100 initial designs X_0, build the other 3 subsets from it and compute the criteria described above.

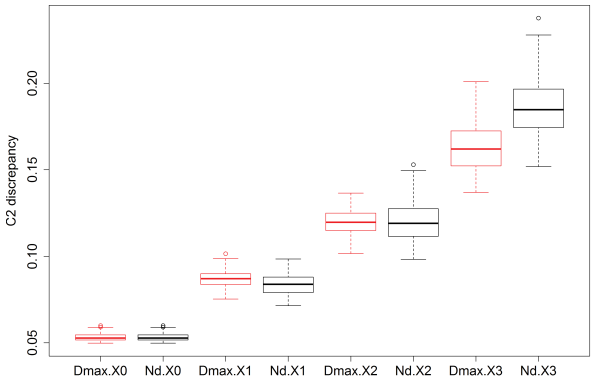
We use $Dmax$(Determinant to be maximized) to denote the designs built by our method and Nd(Nested designs) those built by selecting the points at random.

Figure 4.5a shows the boxplots of the mean size of the edges of the MST. The edges are bigger on the first two nested designs X_1 and X_2. In the last design X_3 the edges are significantly bigger for the $Dmax$ one.



(a) Boxplots of the mean size of the edges of the MST for the different nested designs.

(b) Boxplots of the variance of the size of the edges of the MST for the each of the nested designs.



(c) Discrepancy between the distribution of the points of the design and uniform distribution.

Figure 4.5: Analysis results of Nd on black and $Dmax$ on red.

All the boxplots in figure 4.5b for variances of the size of the edges are smaller for the $Dmax$ designs. Once again the most notable difference is between the two last designs.

The discrepancies in figure 4.5c are similar except, once again, for the

last level. We can conclude that the $Dmax$ design on the highest fidelity level is better. In the other two levels the two methods are very similar.

Since the highest fidelity data is the hardest to obtain we prioritize a good sampling design X_3. We use the $Dmax$ designs to sample the the mean gas speed and mean water level using the three discretizations.

4.4.2 The outputs

The outputs are the mean over a period of 30 seconds of several quantities measured at 9.1 m from the inlet as done in [12, 21]. These quantities are the mean

- gas and water speed G_j and W_j;
- water level L_j.

We use index j to denote an output produced using the j^{th} mesh. We sample

- G_1, W_1 and L_1 over X_1 and X_{test};
- G_2, W_2 and L_2 over X_2 and X_{test};
- and G_3, W_3 and L_3 over X_3 and X_{test};

so that we have a training and a test set for each level and output. We want to predict the outputs of mesh 3 on the test set. The other two levels are considered approximations ordered by the mesh level.

The boxplots of the outputs for the third mesh are shown in figure 4.6.

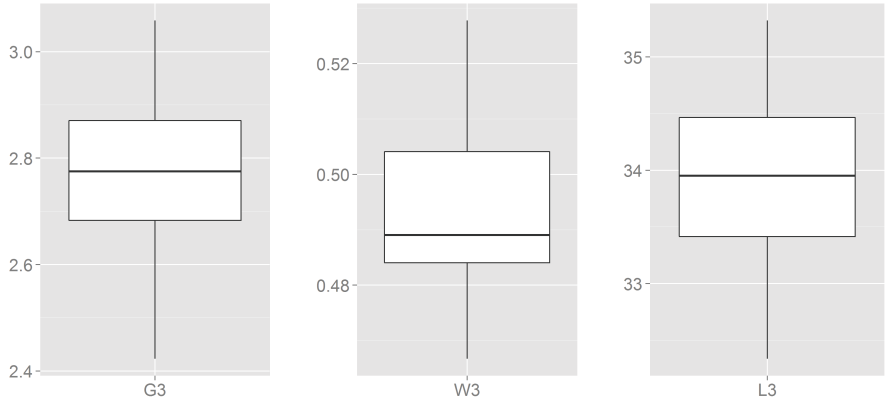


Figure 4.6: Boxplots of the outputs of level 3 sampled over X_0.

We consider three types of models, based on Gaussian processes, for the data. The first model uses data produced only by level 3.

$$(K): \ Y_{3,x} \sim GP(m(x), \sigma_3^2 \gamma(x, x'; \theta_3)).$$

The second is a multi-fidelity model defined recursively as

$$\begin{aligned}(CK): \ Y_{k+1,x} &= b_0 \, Y_{k,x} + Y_{d_k,x}, \quad k \in \{1, 2\}, \\ Y_{d_k,x} &\sim GP(m(x), \sigma_{d_k}^2 \gamma(x, x'; \theta_{d_k})), \\ Y_{1,x} &\sim GP(m(x), \sigma_1^2 \gamma(x, x'; \theta_1))\end{aligned}$$

where $Y_{d_l,x}$ is independent of $Y_{l,x}$ and the relationship between two successive model is linear. Finally, the third model we consider is

$$\begin{aligned}(npCK): \ Y_{k+1,x} &= \varphi(Y_{k,x}) + Y_{d_k,x}, \quad k \in \{1, 2\}, \\ Y_{d_k,x} &\sim GP(m(x), \sigma_{d_k}^2 \gamma(x, x'; \theta_{d_k})), \\ Y_{1,x} &\sim GP(m(x), \sigma_1^2 \gamma(x, x'; \theta_1)).\end{aligned}$$

In this model the relationship between two successive levels is unknown and we estimate it using locally linear polynomials.

For the multi-fidelity models we consider two additional combinations of data sets besides the one with observations sampled from $X_3 \subset X_2 \subset X_1$. One in which we use the training sets $X_3 \subset X_1$ and another in which we

use $X_3 \subset X_2$. The target is always the third level. The fidelity of the others is ordered by the mesh number.

For all the Gaussian processes, we use the same covariance structure: a product of 1-dimensional correlation functions with a single variance parameter and a different covariance parameter for each one of the inputs. Since we have 7 parameters the covariance is written as

$$\gamma(x, x'; \theta) = \prod_{d=1}^{7} \gamma_d(x^d, x'^d; \theta^d)$$

The one dimensional correlations $\gamma_d(x^d, x'^d; \theta^d)$ are Matern 3/2 functions. We use them since we are uncertain about the regularity of the outputs.

4.4.3 Prediction results and estimated parameters

Prediction error and relationship between levels

Table 4.4 shows the $RMSE := \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widehat{y}(x_i) - y_i)^2}$, on the test set for each model. There is no $RMSE$ for K on the columns with more than one data set and no $RMSE$ for the multi-fidelity models on the single data set column. Using multiple data sets is impossible for K and using a single data set to train a multi-fidelity model would produce the same result as K.

For the gas speed G we can see that the first column of table 4.4 contains the biggest $RMSE$s for the $npCK$ multi-fidelity model. Conversely, the CK has its lowest $RMSE$ on this combination. It seems that the data generated by the mesh 1 can be incorporated by the CK model but not by the $npCK$.

Figure 4.7a show the plots of the gas speed of two successive levels. We can see that the relationship $G_1 - G_3$ is not very well defined. On

$Model$	$G_1,\ G_3$	$G_2,\ G_3$	$G_1,\ G_2,\ G_3$	G_3
K	-	-	-	0.09958
CK	0.09871	0.09904	0.09905	-
$npCK$	0.1246	0.08994	0.1137	-
$Model$	$W_1,\ W_3$	$W_2,\ W_3$	$W_1,\ W_2,\ W_3$	W_3
K	-	-	-	0.0157
CK	0.01541	0.01567	0.01555	-
$npCK$	0.01394	0.01476	0.01558	-
$Model$	$L_1,\ L_3$	$L_2,\ L_3$	$L_1,\ L_2,\ L_3$	L_3
K	-	-	-	0.7595
CK	0.8883	0.7873	0.7873	-
$npCK$	0.8289	0.7573	0.7787	-

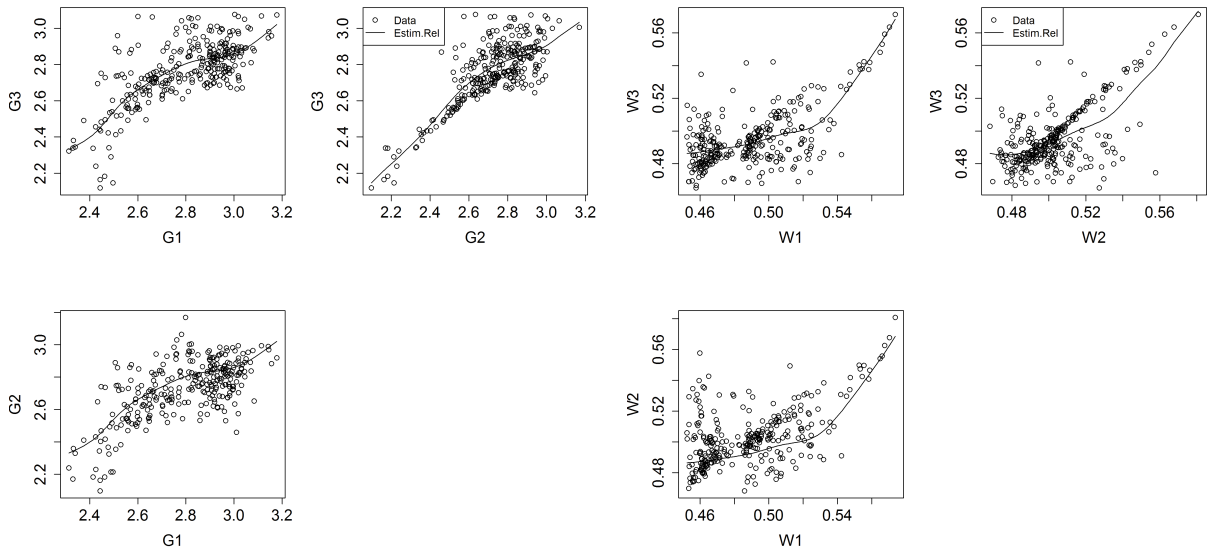
Table 4.4: RMSE on the test set. Each row correspond to a model and each colunm to the training sets used to make the prediction. We used a $Matern$ 3/2 covariance function.

the other hand, the plots involving levels 2 and 3 are less spread. We can see a linear trend on first part of gas speed. This is reflected on the $RMSE$ too: the $npCK$ multi-fidelity model has a lower $RMSE$ for this combination. Although the relationships seems linear, the $RMSE$ of the parametric relationship model CK is similar to that of K.

Something similar can be said about the water speed W. The only difference is that the $npCK$ model can use the data of level 1 to reduce the $RMSE$. We can see in figure 4.7b that the relationship between the outputs is nonlinear.

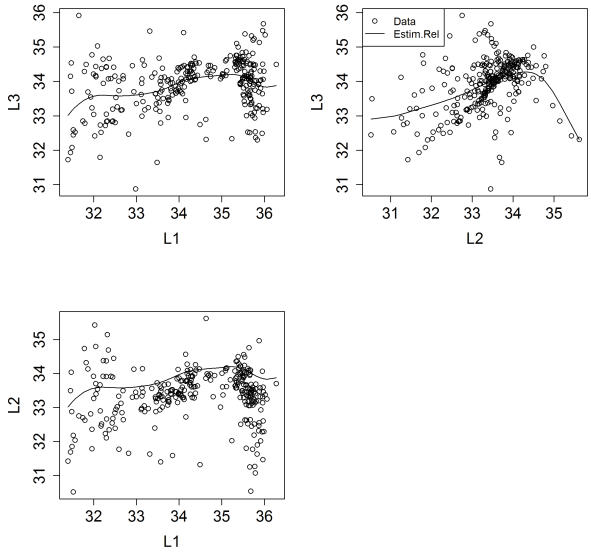
It seems that for the water level L, the best option is the one data set model K or $npCK$. By looking at figure 4.7c we can conclude that there is not a clear relationship between the data of different levels. Nonetheless, both multi-fidelity models have a comparable performance when trained with data from levels 2 and 3.

More generally, the $RMSE$ of the predictions built using all three levels is not better that the ones made by only using levels 2 and 3. In fact, it deteriorates the prediction of the $npCK$ model for the three outputs. This



(a) Gas speed.

(b) Water speed.



(c) Water level.

Figure 4.7: Observations of two successive levels with a relationship estimated by locally linear polynomials.

is coherent with the other results. In any case, the $npCK$ model represents an improvement on the gas and water speeds predictions and gives results for the water level that are comparable with the other two models.

An important conclusion of this part is that considering the approximations 1 and 2 to study the influence of the parameters on the water level does not represents a real improvement. The target output seems to have a

more complex relationship with its approximations than the ones modeled by Ck and $npCK$.

Using level 2 does improves the prediction error for the gas and water speeds made by $npCK$.

We should remark that the different meshes were obtained by refining different dimensions. From $50 \times 20 \times 1$ to $50 \times 40 \times 1$ we doubled the cells in the y direction whereas for the $50 \times 20 \times 1$ to $100 \times 40 \times 1$ we refined the x. After the analysis we did above, we could suspect that refining on the y dimension is more important.

Finally, by looking back, we can think of each output as an example of three different multi-fidelity settings. For G the linear relationship gives good results; for W it is not sufficient to assume a linear relationship and estimating the relationship improves the error and finally for L it is difficult to use the approximate data.

Estimated parameters

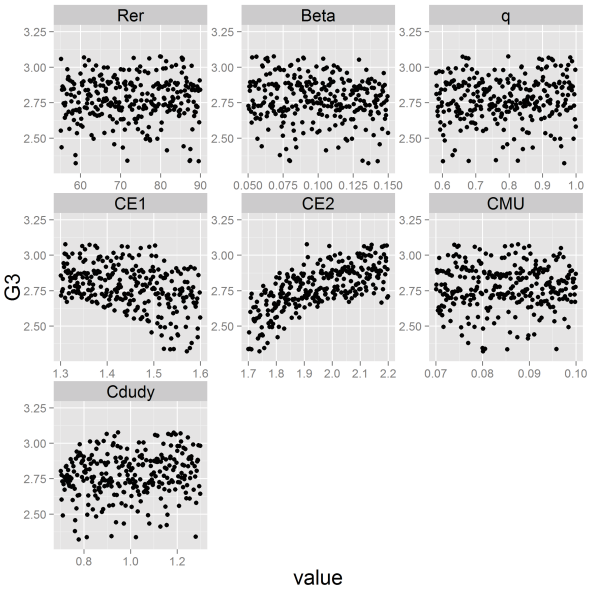
We briefly described the form of the relationship between two successive levels of fidelity. Here we discuss the choice of parameters in more detail.

First we make a remark related to the CK model. Using the more complex relationship

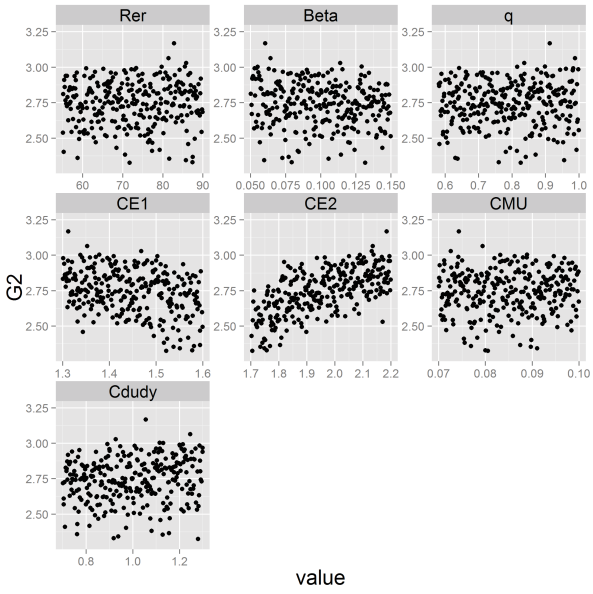
$$(CK(x)): \ Y_{k+1,x} = (b_0 + b_1^T x)Y_{k,x} + Y_{d_k,x}$$

yields very similar results. We keep CK since it is a more simple model in terms of parameters.

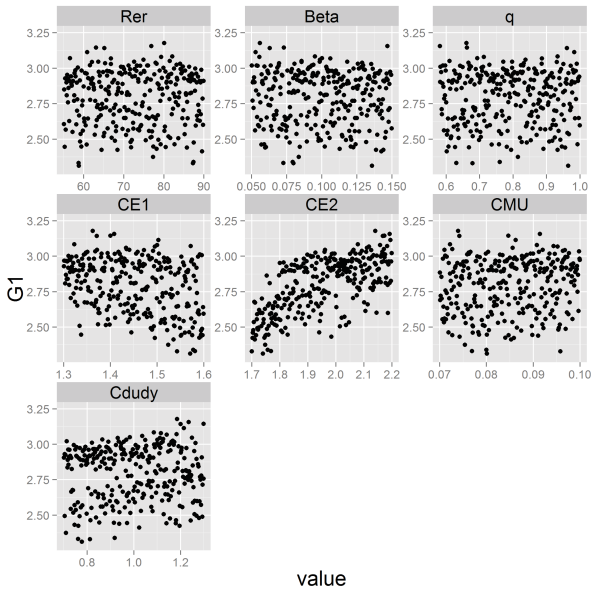
For the mean of all the $Y_{l,x}$ and the all difference processes $Y_{d_l,x}$ we use a linear combination of the inputs like $\alpha_0 + \alpha_1^T x$. So for example, for level one $E[Y_{1,x}] = \alpha_0 + \alpha_{1,1}x_1 + \cdots + \alpha_{1,7}x_7$ has 8 mean parameters.



(a) Mean gas speed G_3.



(b) Mean gas speed G_2.



(c) Mean gas speed G_1.

Figure 4.8: Projections of the mean gas speed on the outputs on each dimension of the inputs.

We have one covariance parameter for each one of the inputs. As we discussed on chapter 2, we can interpret each one of the parameters as a measure of relevance of each dimension. A big covariance parameter implies small variations along the input it represents.

We focus on the multi-fidelity models using levels 2 and 3. They are

$$
\begin{aligned}
(CK): \; Y_{3,x} &= \beta_0 Y_{2,x} + Y_{d_2,x}, \\
Y_{d_2,x} &\sim GP(\alpha_0 + \alpha_1^T x, \sigma_{d_2}^2 \gamma(x, x'; \theta_{d_2})), \\
Y_{2,x} &\sim GP(\alpha_0 + \alpha_1^T x, \sigma_2^2 \gamma(x, x'; \theta_2))
\end{aligned}
$$

and

$$
\begin{aligned}
(npCK): \; Y_{3,x} &= \varphi(Y_{2,x}) + Y_{d_2,x}, \\
Y_{d_2,x} &\sim GP(\alpha_0 + \alpha_1^T x, \sigma_{d_2}^2 \gamma(x, x'; \theta_{d_2})), \\
Y_{2,x} &\sim GP(\alpha_0 + \alpha_1^T x, \sigma_2^2 \gamma(x, x'; \theta_2)).
\end{aligned}
$$

Table 4.5 shows the estimated parameters for G for all the three models.

Model Param.	Re_r	β_1	q	$CE1$	$CE2$	CMU	$Cdudy$
$K\ \theta_3$	69.10	0.1940	0.8240	0.1400	0.1610	0.0598	0.7840
$CK\ \theta_2$	53.30	0.0103	0.0394	0.5940	0.2220	0.0598	0.0244
$npCK\ \theta_2$	69.10	0.0673	0.8240	0.0450	0.1240	0.0598	1.1600
	Re_r	β_1	$Ceq32$	$CE1$	$CE2$	CMU	$Cdudy$
$CK\ \theta_{d_2}$	67.4000	0.1900	0.4620	0.4010	0.0721	0.0575	1.1600
$npCK\ \theta_{d_2}$	53.5	$10e^{-10}$	0.705	0.233	$10e^{-10}$	0.0165	0.639
Interval size	35	0.05	0.4	0.3	0.5	0.03	0.6

Table 4.5: Estimated covariance parameters for the gas speed G. The multi-fidelity models use levels 2 and 3. The last row contains the size of the intervals of the parameters.

For the gas speed, the parameters estimated by the $npCK$ model represent the variation of the outputs along each input dimension shown in figure 4.8b. All are big parameters with respect to their corresponding ranges, shown on the last row of table 4.5, except for $CE1$ and $CE2$. This is not the case for CK: β_1, q and $Cdudy$ are small.

On the other hand $npCK$ sets to zero the covariance parameters of β_1 and $CE2$ of the difference process. The variations along Re_r, q and $Cdudy$ are small. For $CE2$ we expect to have a difference process that varies a lot for the two models.

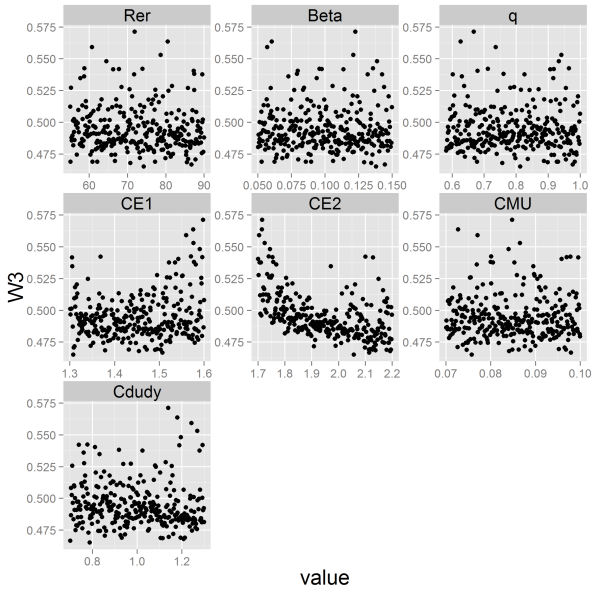
Model Param.	Re_r	β_1	q	$CE1$	$CE2$	CMU	$Cdudy$
$K\ \theta_3$	67.40	0.1900	0.8180	0.0838	0.1060	0.0575	1.1600
$CK\ \theta_2$	0.751	0.1940	0.8240	0.3900	0.9880	0.00836	1.1600
$npCK\ \theta_2$	54.90	0.1940	0.8240	0.0692	0.1250	0.0598	0.9940
	Re_r	β_1	$Ceq32$	$CE1$	$CE2$	CMU	$Cdudy$
$CK\ \theta_{d_2}$	59.90	0.1900	0.4620	0.4010	0.0721	0.0575	1.1600
$npCK\ \theta_{d_2}$	28.9	$10e^{-10}$	0.809	0.456	0.478	$1e-10$	0.491
Interval size	35	0.05	0.4	0.3	0.5	0.03	0.6

Table 4.6: Estimated covariance parameters for W. The multi-fidelity models use levels 2 and 3. The last row contains the size of the intervals of the parameters.

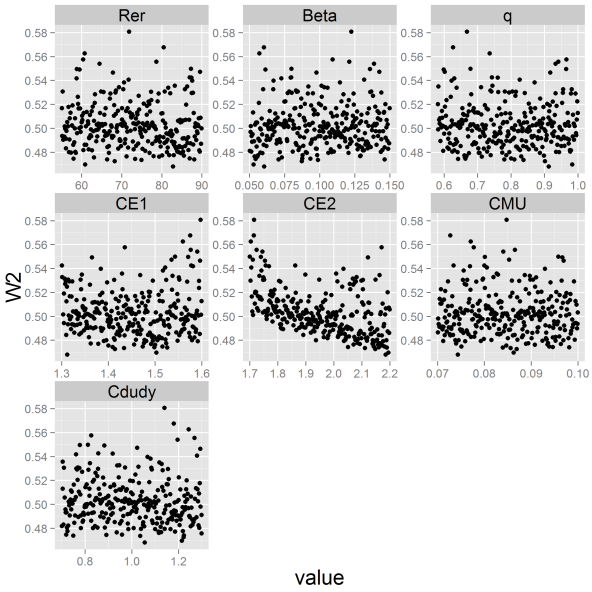
The estimated parameters for the water speed are shown in table 4.6 and the projection plots in 4.9. This is the case in which the relationship between levels 2 and 3 is nonlinear. Once again it can be seen in figure 4.9b that $CE1$ and $CE2$ are the parameters for which the outputs vary the most. This is reflected on the estimated coefficients of $npCK$.

Finally, for the water level we have table 4.7 and figure 4.10. In the figure we can see that the influence of $CE1$ and $CE2$ is not as clear as for the other two outputs. The multi-fidelity models argue that the difference process between levels 2 and 3 has a high variability in the $CE1$ and $CE2$ directions.

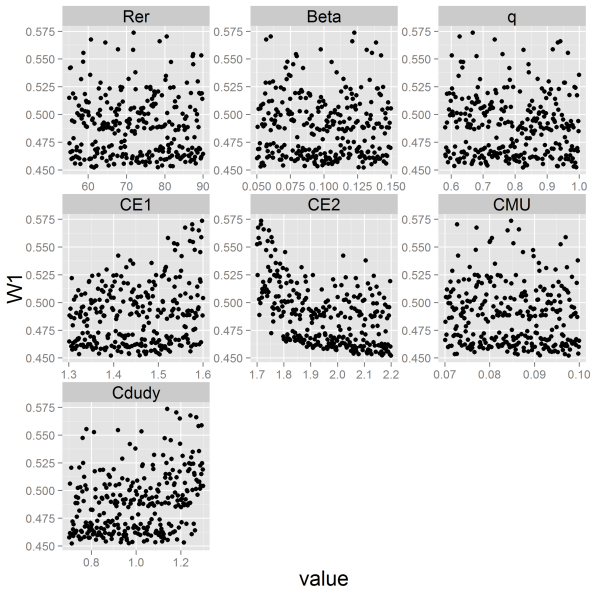
In the next section we compute the Sobol indexes on the metamodels we studied here.



(a) Mean water speed W_3.



(b) Mean water speed W_2.



(c) Mean water speed W_1.

Figure 4.9: Projections of the mean water speed W on the outputs on each dimension of the inputs.

4.4.4 Sensitivity analysis

We already argued that the actual value of the parameters we used as inputs are uncertain. Here we explore the sensitivity of the output to variations of the parameters. The objective is to classify the parameters in function of their influence on the outputs. Our main reference for this section is the PHD thesis of Alexandre Janon [28].

$Model\ Param.$	Re_r	β_1	q	$CE1$	$CE2$	CMU	$Cdudy$
$K\ \theta_3$	67.4	0.1540	0.8180	0.0661	0.0697	0.0575	1.160
$CK\ \theta_2$	25.1	$1e-10$	$1e-10$	0.594	0.221	$1e-10$	0.153
$npCK\ \theta_2$	69.1	0.0730	0.8240	0.0392	0.1240	0.0205	0.994
	Re_r	β_1	$Ceq32$	$CE1$	$CE2$	CMU	$Cdudy$
$CK\ \theta_{d_2}$	67.40	0.1540	0.8180	0.0661	0.0698	0.0575	1.160
$npCK\ \theta_{d_2}$	66.90	0.0863	0.8180	0.0731	0.0582	0.0136	1.160
Interval size	35	0.05	0.4	0.3	0.5	0.03	0.6

Table 4.7: Estimated covariance parameters for L. The multi-fidelity models use levels 2 and 3. The last row contains the size of the intervals of the parameters.

The method we use models the parameters by using a probability law. This law together with the model or meta-model of the air-water flow determines the distribution of the outputs.

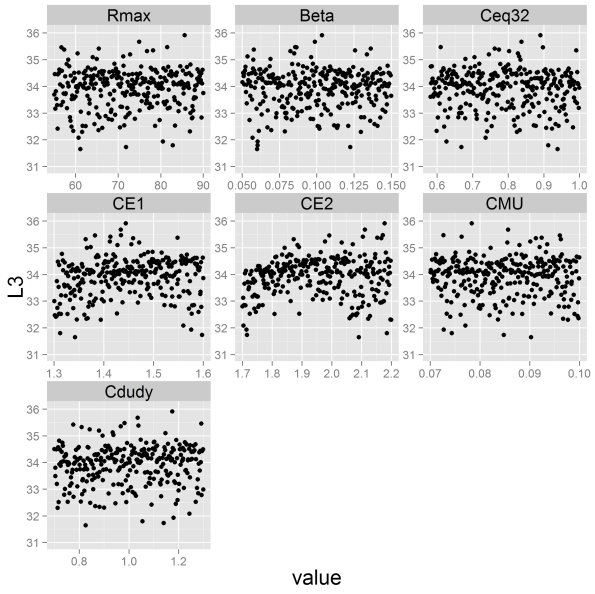
To classify the parameters we work with the Sobol indexes. The idea is to determine which parameters produce most of the uncertainty on the output so that we can focus only on them by setting the others to a nominal value.

Before defining the Sobol indexes we assume that the parameters, that we note $X_1^P, ..., X_7^P$, are independent and that the outputs are such that $E[G_k^2] < +\infty$, $E[W_k^2] < +\infty$ and $E[L_k^2] < +\infty$ for $k \in \{1, 2, 3\}$.

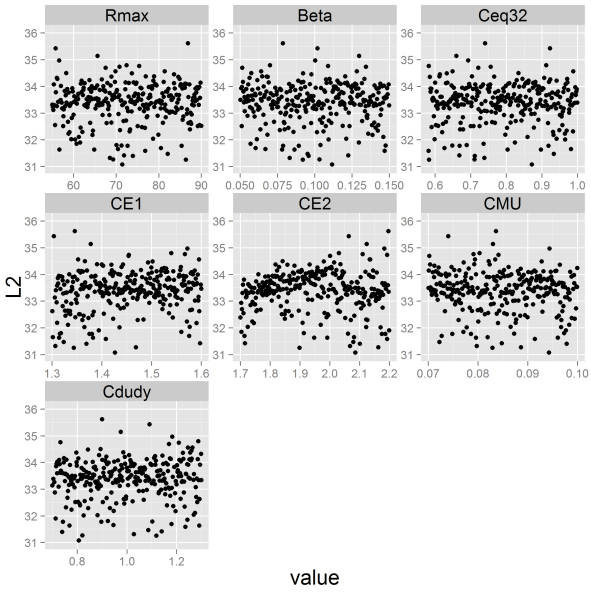
The Sobol index for X_i is defined by considering the variance of $E[G_k|X_i^P]$. Recall that $E[G_k|X_i^P]$ is the best approximation of G_k among all functions of X_i. The bigger this variance is, the more influential on the output X_i is. The index is

$$S_i = \frac{Var(E[G_k|X_i^P])}{Var(G_k)}.$$

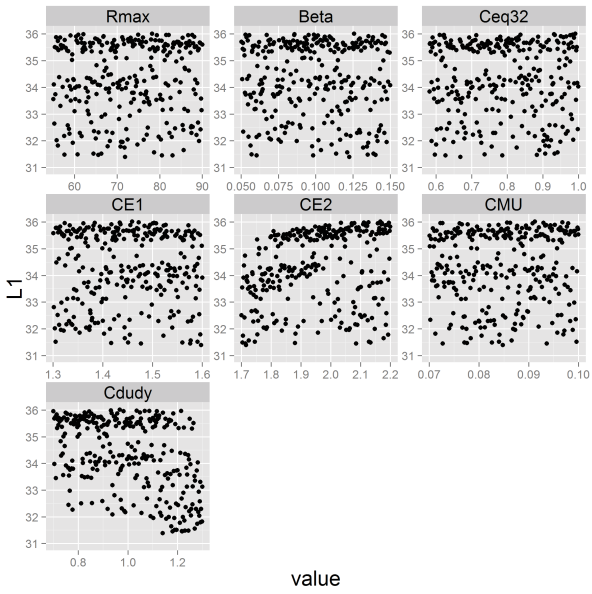
It represents the fraction of the total variation attributed to X_i. For a



(a) Mean water level L_3.



(b) Mean water level L_2.



(c) Mean water level L_1.

Figure 4.10: Projections of the mean water level L on the outputs on each dimension of the inputs.

subset of $u \subset \{1, ..., 7\}$

$$S_u^{cl} = \frac{Var(E[G_k|X_i^P, i \in u])}{Var(G_k)}.$$

quantifies the influence of all the parameters indexed by u. The total

indexes are

$$S_i^T = 1 - S^{cl}_{\{1,\dots,7\}\setminus\{i\}}.$$

They represent the effect of the i^{th} variable and its interactions with all the other variables.

The indexes are computed using Monte-Carlo integrations. In our case, we cannot compute enough samples and thus we cannot compute the indexes directly. We estimate the Sobol indexes on the output of the meta-models we considered before.

More precisely, we compute the indexes on the four meta-models. Two multi-fidelity models CK and $npCK$ that use levels 2 and 3 of responses. A single level regression model $K(X_1)$ that we train using 210 observations of gas speed and water level over X_1 produced by the simulator 3 and the last one is $K(X_3)$, a one data set model sampled over X_3.

The model with the lowest $RMSE$ is $K(X_1)$ for each one of the outputs. It is 0.08982 for G_3; 0.01382 for W_3 and 0.6694 for L_3. The other $RMSE$s are shown in table 4.4. This model is our reference.

Figures 4.11, 4.12 and 4.13 show the Sobol indexes computed using the four meta-models.

The most influential parameters for the three outputs are $CE1$ and $CE2$. In the gas speed the influence is due mainly to main effects with some interactions while in the water speed and level the interactions have a more important role.

For the gas speed, the predictions of the two multi-fidelity models give the same general picture as $K(X_1)$. They both attribute the influence of $CE1$ and $CE2$ to main effects and some interactions. In the two multi-fidelity models we can see that the first variable Re_r has a main effect on the output that is not present on the $K(X_1)$ model.

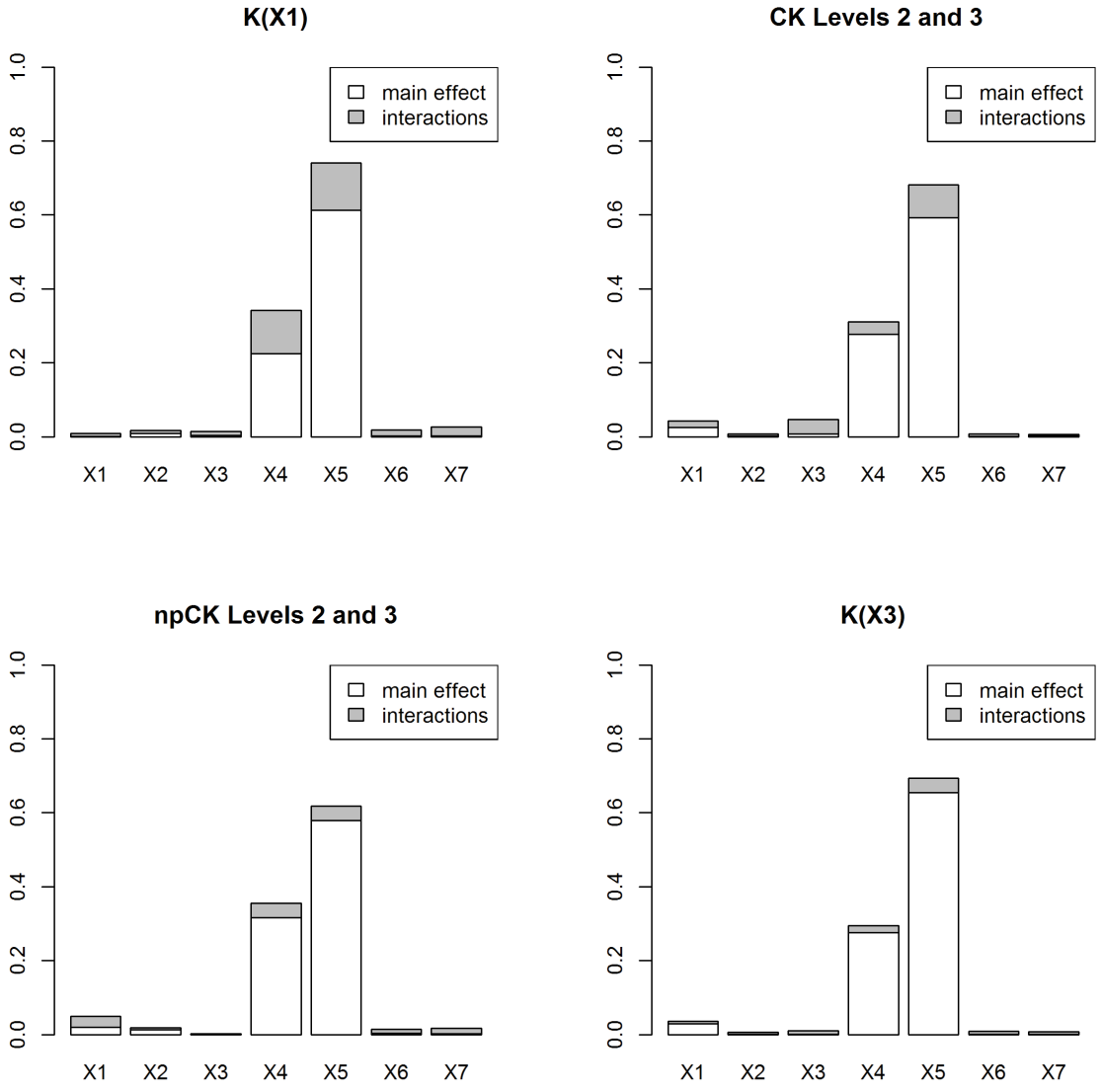


Figure 4.11: Sobol indexes for the gas speed G_3 computed using FAST. The X_i variables are Re_r, β_1, q, $CE1$, $CE2$, CMU and $Cdudy$.

For the water level, the index corresponding to β_1 and $Cdudy$ have a main effect for CK and $npCK$ that is not present on the reference. Besides these indexes, $npCK$ captures the form of the other indexes.

Finally, in the water level, the $K(X_3)$ model produces a result similar to that of $K(X_1)$ for the two most influential variables. For the other variables the estimation of $npCK$ is closer to that of $K(X_1)$. Specially for the last two indexes.

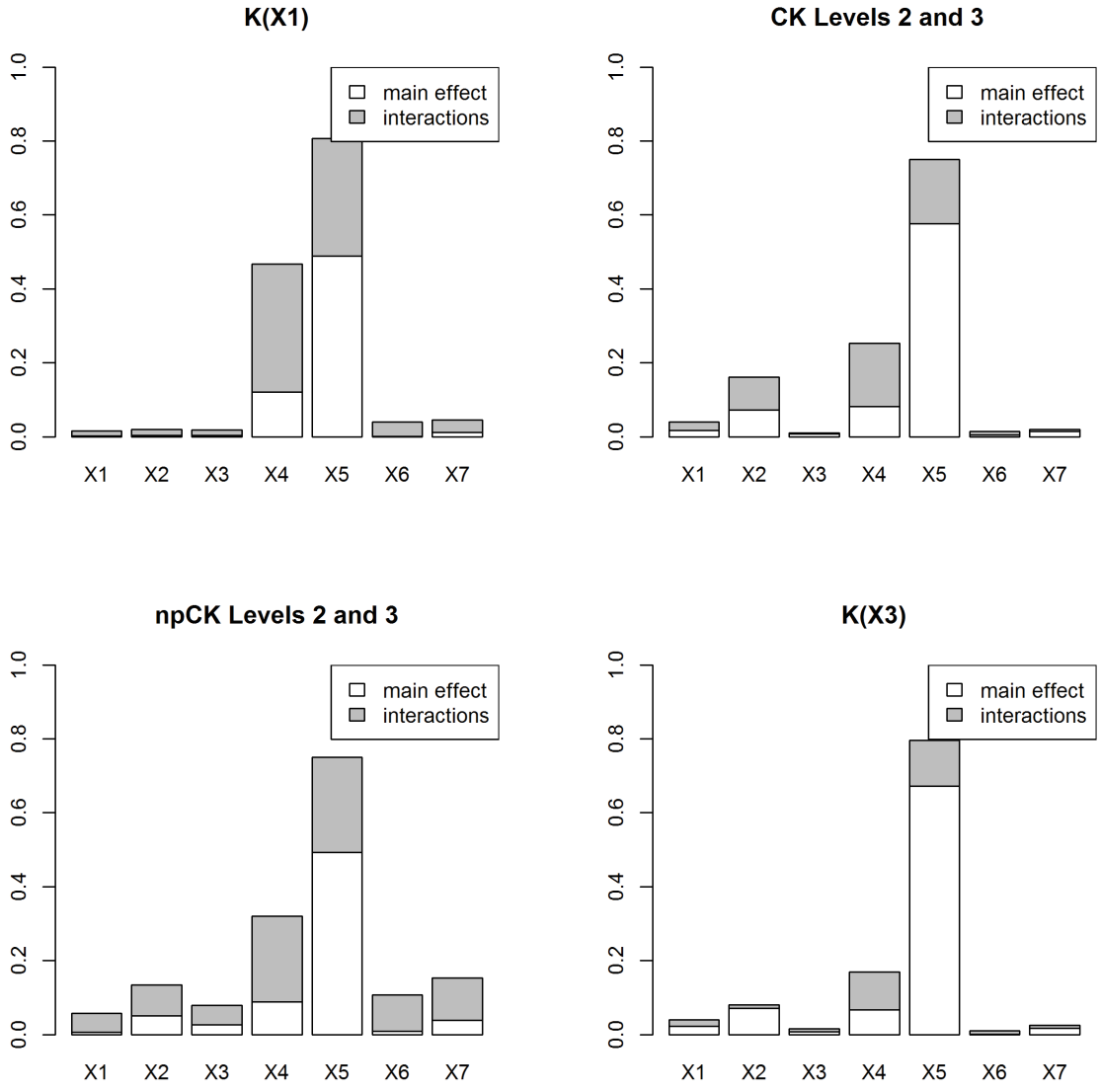


Figure 4.12: Sobol indexes for the water speed W_3 computed using FAST. The X_i variables are Re_r, β_1, q, $CE1$, $CE2$, CMU and $Cdudy$.

Figure 4.10 show the projections of the water level into each one of the dimensions. There is not a clear tendency in any of the plots. This along the sensitivity analysis results suggests that most of the explained variability is due to the interactions.

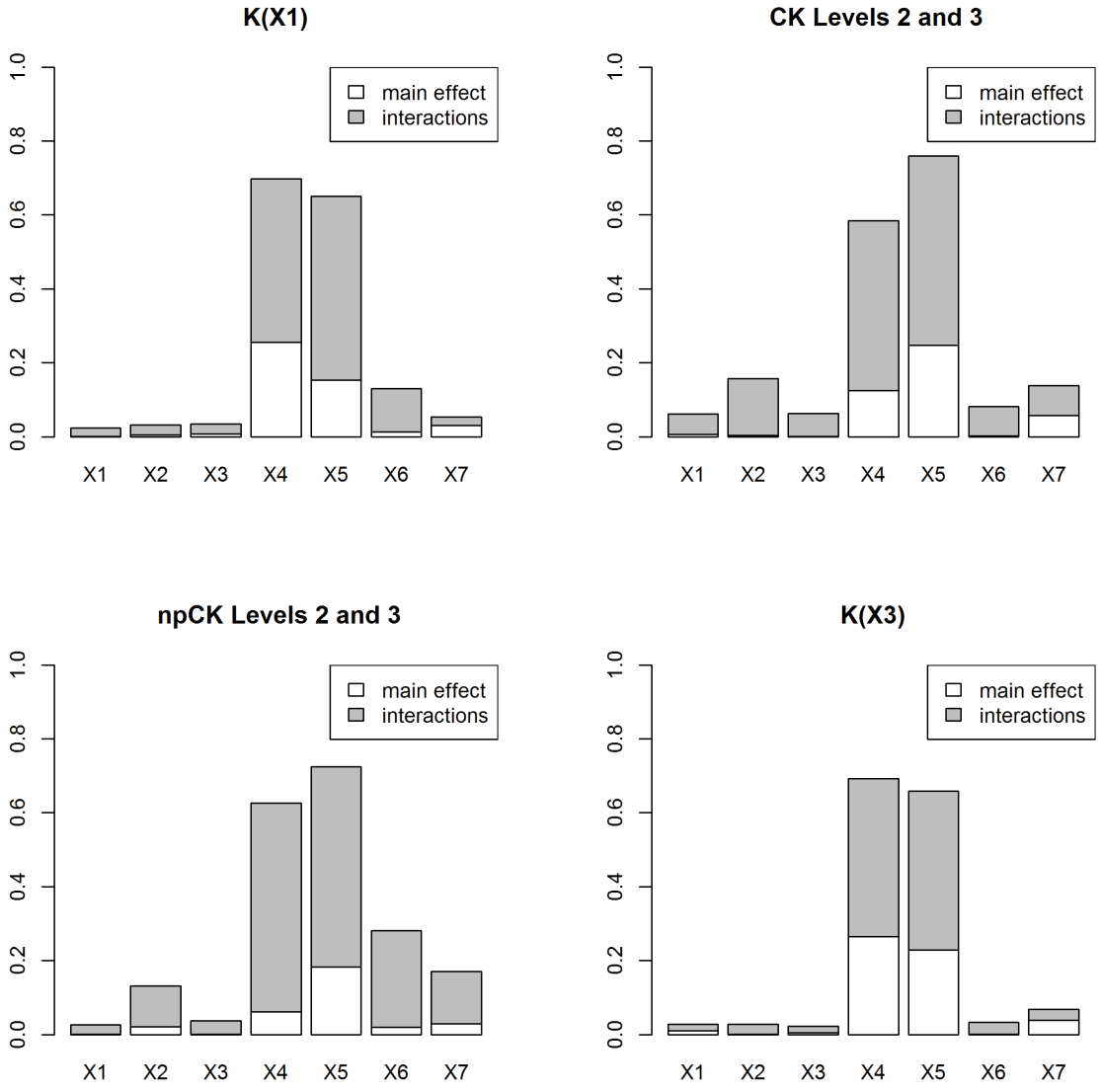


Figure 4.13: Sobol indexes for the water level L_3 computed using FAST. The X_i variables are Re_r, β_1, q, $CE1$, $CE2$, CMU and $Cdudy$.

4.5 Conclusion

A practical multi-fidelity case study. In this chapter we presented a diphasic air-water flow case study motivated by the study of the aging of nuclear power plants. In this case study we dispose of three different configurations that defined three levels of fidelity with an order.

The case is described in detail with references and the parameters are selected by an expert engineer.

We study three different outputs G, W and L. Each one represents a case of multi-fidelity data. On the first a linear relationship gives good results. For the water speed W, estimating the relationship is useful. Finally, L is an example in which it is difficult to use the approximate data to improve the prediction.

Nested designs. In order to build the multi-fidelity models we introduce an algorithm to build nested designs. The algorithm selects subsets that maximize the entropy of the outputs.

Testing multi-fidelity models. By using the multi-fidelity models we were able to see that approximation 1 was not useful to study the influence of the parameters on the target outputs produced by the mesh 3.

We see that for the gas and water speed, estimating the relationship reduced the test error.

The predictions regarding the water level are difficult. In this case it seems preferable to use the target level directly. The CK model produced results similar to those of the one data set model K.

According to the Sobol indices, the most influential parameters were $CE1$ and $CE2$. This means that the most important part of the model is related to the turbulence models of each one of the fluids.

In the gas speed, the influence of these parameters is due to the main effects. For the water speed, the influence of $CE1$ is due to main effect while that of $CE2$ is due to interactions. For the water level it is due principally to interactions.

It seems that to improve the predictions of the water level we need to change the covariance structure. Instead of using a product of one dimensional it would be advisable to try an ANOVA like kernel to look for higher order interactions - see [17, 18].

In the next chapter we present a different regression model that is based on an adaptive wavelet decomposition.

Chapter 5

Adaptive Wavelets and Multi-fidelity

In this chapter we study an alternative method to make a prediction over an unobserved input in the multi-fidelity context. Instead, of using Gaussian processes, we consider a wavelet decomposition to study the different unknown functions. We explore these functions by constructing a hierarchy of approximations with increasing level of detail, proceeding from coarsest to finest resolution. This allows for an efficient selection of the degrees of freedom of the problem. Additionally, while building these approximations we explore the zones in which the function varies the most with the idea of eventually adding observations at precise locations when needed.

More precisely, the idea of the method we present here is to come up with an approximation of a function of the form

$$f := \sum_{i \in I} d_i \; w_i$$

where the w_i are wavelets; d_i are unknown coefficients and I is a set of indexes selected by using all the data

First, we start by introducing the algorithm we use to select the index set I and to compute the coefficients d_j for one data set. It is an iterative procedure with two main steps: counting the number of points in the domain of each basis function to avoid over fitting, and selecting the

estimated coefficients to explore the data locally and avoid redundancy.

Then, we present its multi-fidelity counterpart in which we apply the the iterative procedure sequentially to each data set and the difference between two successive fidelity levels. We apply this algorithm to a numerical example at the end of the chapter.

5.1 Adaptive wavelet decomposition

In this section we introduce the algorithm proposed by Daniel Castaño in his PHD thesis [8]. This is the one data set version of our multi-fidelity algorithm.

Here we are given data set $(x_1, y_{x_1}), ..., (x_n, y_{x_n})$. We assume that this data was sampled from an unknown function $f : [0, 1] \to \mathbf{R}$ that can be written as

$$\begin{aligned} f(x) &= c\ \phi(x) + \sum_{j=0}^{\infty} \sum_{k=0}^{2^j-1} d_{j,k}\ w(2^j x - k) \\ &=: c\ \phi(x) + \sum_{j=0}^{\infty} \sum_{k=0}^{2^j-1} d_{j,k}\ w_{j,k}(x). \end{aligned}$$

for any $x \in [0, 1]$. Where ϕ represents the mean or low frequency behavior of f and the $w_{j,k}$'s account for the local variations - or local high frequency behavior - of the unknown function. The function ϕ is called the scaling function and w is called a wavelet. The wavelets we study are compactly supported wiggles. The resolutions index j is a measure of how local the wiggle is and k represents its location.

The algorithm is a coarse to fine approximation. At each step we analyze the decomposition up to a resolution j and add the wavelets at resolution $j + 1$ that contain enough data points and that represent a significant contribution. If we note K_j the set of wavelets at resolution j used in the

decomposition, we want to build an approximation of the form

$$f(x) \approx \widehat{c}\,\phi(x) + \sum_{j=0}^{J} \sum_{k \in K_j} \widehat{d}_{j,k}\; w_{j,k}(x).$$

where $\widehat{c}$ and $\widehat{d}_{j,k}$ are estimated coefficients. This decomposition should not overfit the data and each wavelet should add a significant amount of information.

Usually, a wavelet decomposition is built by using a fine to coarse algorithm described in section 3 of appendix B. This is a very efficient algorithm that works in a particular configuration that requires observations over a dyadic grid and the number of observations determines the maximal resolution. Here we use irregularly sampled observations and we do not know what the highest resolution is, we keep adding wavelets until the algorithm stops.

The algorithm consists in two main steps. Each step has a parameter that controls the selection process. We follow [8] and introduce the first parameter of the algorithm, the number of points q in the support of a wavelet used in the decomposition.

5.1.1 Number of points in the support

The algorithm starts with an initial decomposition that uses all the wavelets up to a resolution j_0

$$c\;\phi(x) + \sum_{j=0}^{j_0} \sum_{k=0}^{2^j-1} d_{j,k}\; w_{j,k}(x).$$

The idea is to add, to the initial decomposition, some wavelets at resolution $j_0 + 1$ to obtain an expression with an extra term like

$$c\ \phi(x) + \sum_{j=0}^{j_0} \sum_{k=0}^{2^j - 1} d_{j,k}\ w_{j,k}(x) + \sum_{k \in K_{j_0+1}} d_{j_0+1,k}\ w_{j_0+1,k}(x)$$

where we note K_{j_0+1} the set that contains the locations of the selected wavelets. The functions we add, at resolution $j_0 + 1$, are the children of the wavelets that contain at least q points in their support. We call $w_{j+1,k}$ and $w_{j+1,k+1}$ the children of $w_{j,k}$. All the wavelets at resolution $j + 1$ are the children of wavelets at resolution j so adding all the children amounts to adding a full resolution to the decomposition - see figure 5.1.

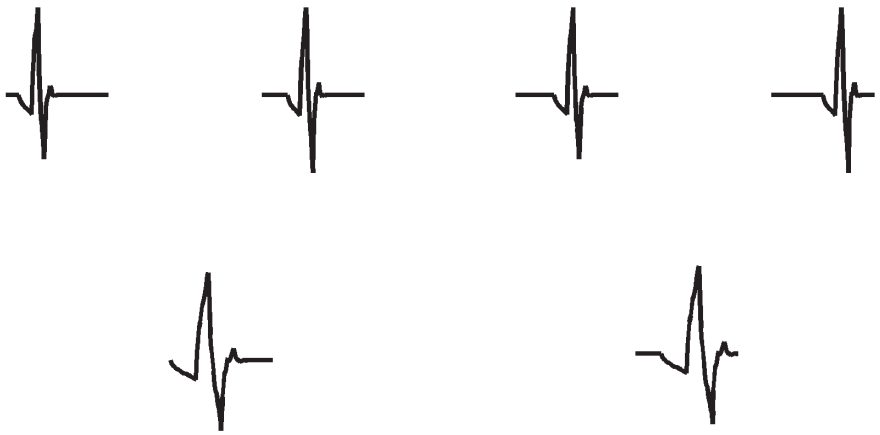


Figure 5.1: On the top row we have the children of the wavelets of the bottom row. Each one has two children and the four children form a complete resoltion. This are Daubechies wavelets.

We repeat this step until there are no more wavelets left with at least q points is their support. To illustrate this, we apply the selection procedure to the example shown in figure 5.2. We use Haar's wavelets defined in appendix B. Higher resolution wavelets are selected in zones where the observations are more dense. We see how q is a parameter that might

control the bias-variance trade off.

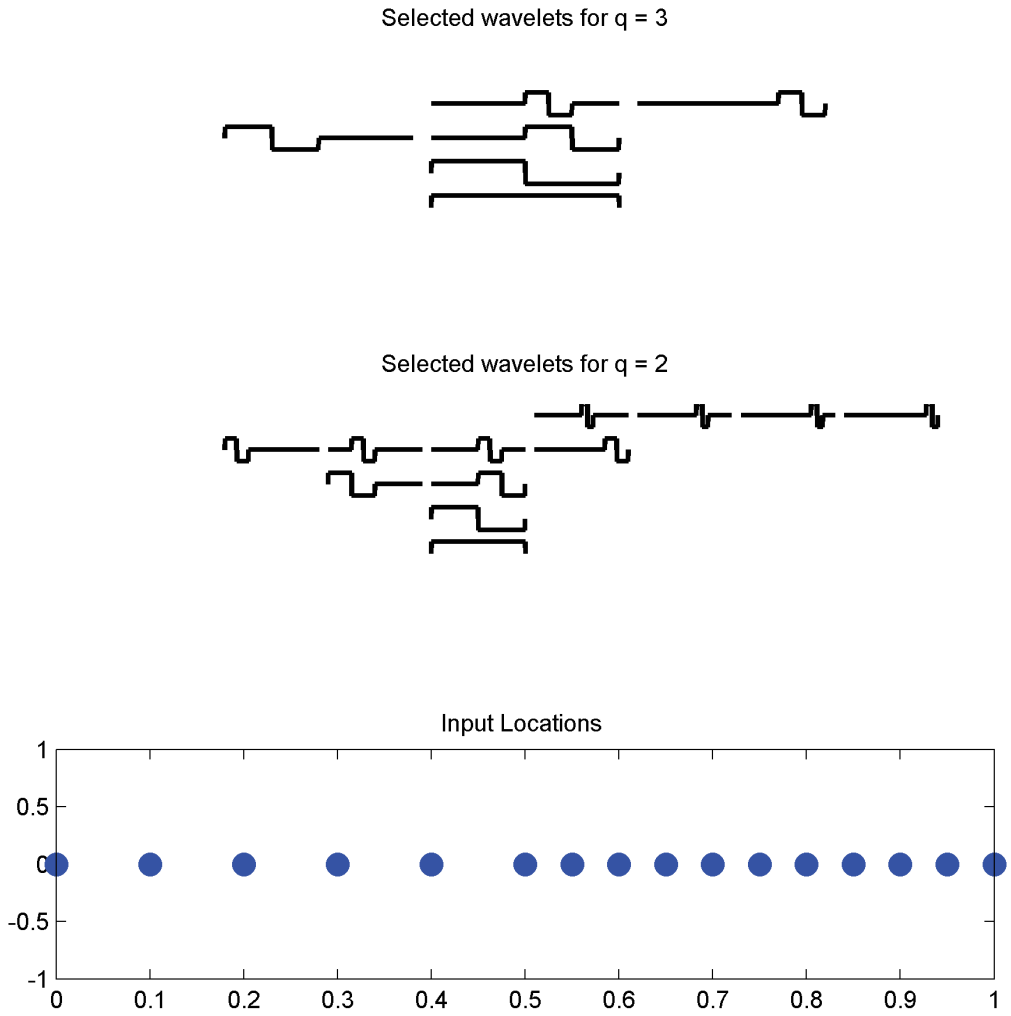


Figure 5.2: The two figures on top represent the selected wavelets. The resolution increases from the bottom to the top. The last figure shows the distribution of the inputs used to select the wavelets. The second part of the domain contains twice as many inputs as the first.

Before moving to the second example, we define the algorithm as:

Algorithm 3 q points in the support.

procedure q POINTS
    Fix j_0 and q.
    Build the first decomposition up to resolution $j_{max} \leftarrow j_0$.
    while $K_{j_{max}} \neq \emptyset$ **do**
        Build $K_{j_{max}+1}$ by choosing the children of $K_{j_{max}}$ with q points in their support.
        $j_{max} \leftarrow j_{max} + 1$.
    return $K_{j_0+1}, \dots, K_{j_{max}}$.

The second example shows the effect of q in the prediction. We sample points from a line. We select the indexes used in the decomposition by using algorithm 3. Then, we estimate the coefficients of the decomposition with the ones that minimize mean square error in the sample and compute the prediction on $[0, 1]$.

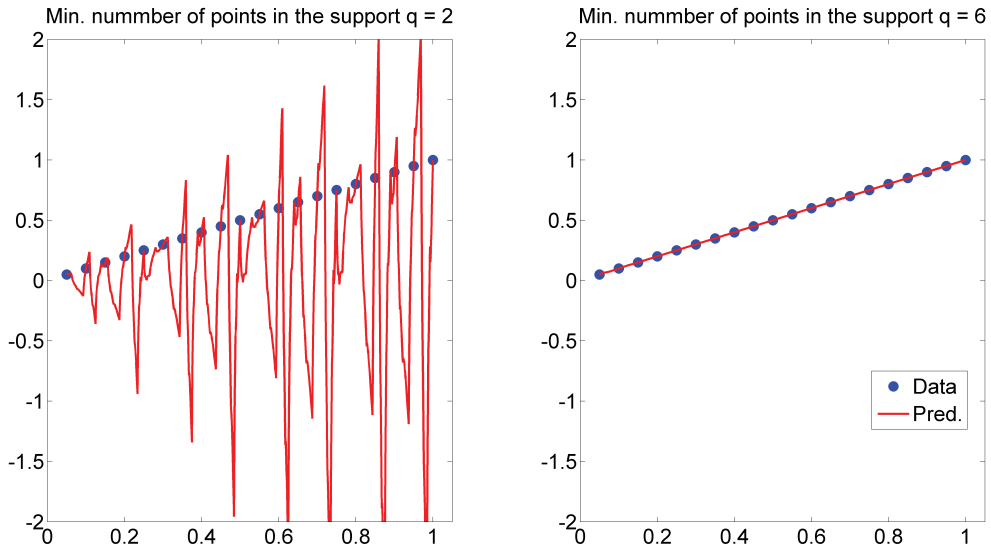


Figure 5.3: The two predictions have 0 error in the sample. On the left the prediction overfits the simple structure of the data. Chossing a bigger q corrects this on the right. We use Daubechies wavelets with 4 degrees of freedom to build the decomposition.

By choosing a correct number of points q, we avoid adding high oscillations where no data is available. This gives a decomposition that is coherent with the spatial distribution of the observations. However, this criterion does not take into account the local regularity of the data. We

can select very high resolution wavelets in places where there are plenty of observations that do not vary much locally. This may lead to a decomposition in which the contribution of many of the selected wavelets is small. In the example above the most important coefficients are those of the scaling function, and the wavelets up to resolution 2. The other coefficients are almost zero as shown in figure 5.4. It seems that the wavelets at higher resolution do not contribute much in the decomposition, they add redundant information. This makes sense, the structure of the data does not present any of the local variations of the wavelets at higher resolution, it is very regular.

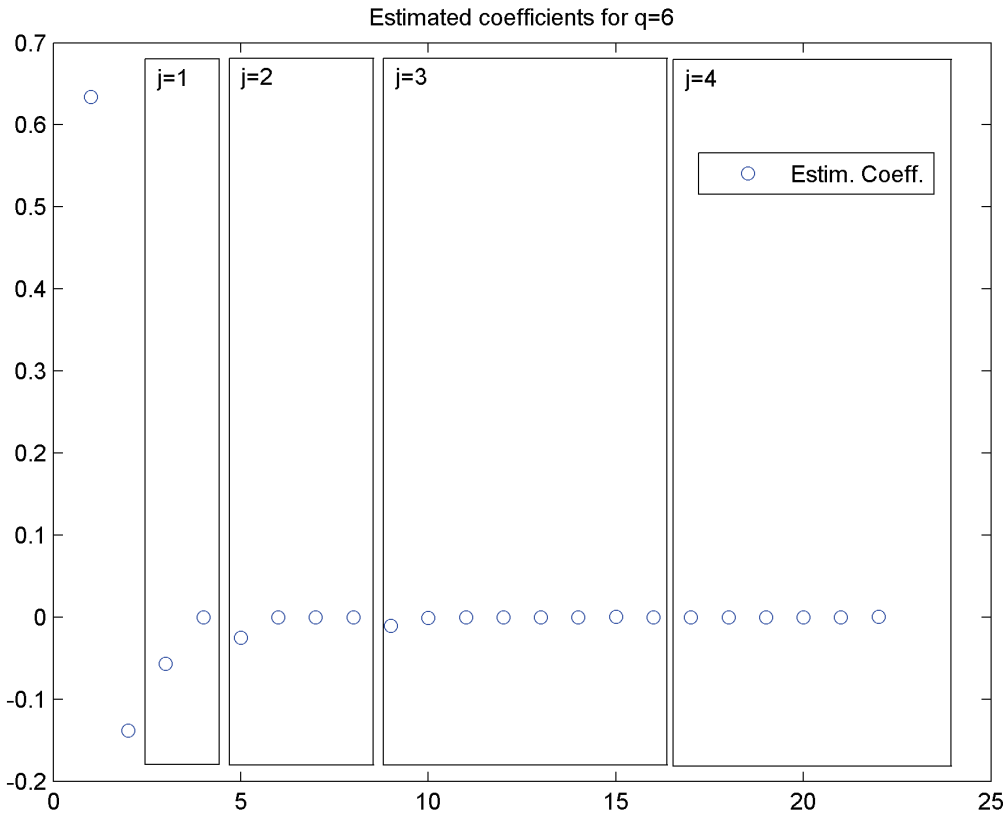


Figure 5.4: Value of the estimated coefficients on the example above with $q = 6$. Each box contains the coefficients of a particualr resolution.

The coefficients in the decomposition represent the influence of its corresponding wavelet. A small coefficient can be interpreted a sign of smoothness at a certain scale and location. On the other hand, a big coefficient

can be interpret as an indicator of potential need for further local refinement. This refinement, can mean adding higher resolution wavelets or new observations at a precise location.

In the second part of the algorithm, by following [8], we introduce the new threshold parameter t_0, to keep the wavelets whose coefficients are big. This parameter is a complement to the number of points q. By using t_0, we take into account the local regularity of the function in the decomposition. This leads to a more efficient and simple representation by wavelets.

5.1.2 Selecting the coefficients

Here we modify algorithm 3. At each step after we select the children in K_{j+1}, we estimate the coefficients of the decomposition by minimizing the mean square error on the data. We make a second selection and keep the children in K_{j+1} whose estimated coefficients are higher in absolute value than a threshold t_0. By doing this we identify zones of regularity where we avoid adding high frequency wavelets and we also recognize other zones with potential need of further local refinement.

The final algorithm is

Algorithm 4 q points in the support and selected coefficients $> t_0$.

procedure q POINTS AND t_0
  Fix j_0 and q.
  Build the first decomposition up to resolution $j_{max} \leftarrow j_0$.
  while $K_{j_{max}} \neq \emptyset$ **do**
    Build $K_{j_{max}+1}$ by choosing the children of $K_{j_{max}}$ with q points in their support.
    Build $\widehat{K}_{j_{max}+1}$ by computing $\widehat{d}_{j+1,k}$ and keeping the wavelets with $|\widehat{d}_{j+1,k}| \geq t_0$.
    $K_{j_{max}+1} \leftarrow \widehat{K}_{j_{max}+1}$ and $j_{max} \leftarrow j_{max} + 1$.
  return $K_{j_0+1}, \dots, K_{j_{max}}$.

In the following example we consider a function with two very regular parts and a jump in the middle as shown on the left panel of figure 5.5. We use Daubechies wavelets on the decomposition. The function is sampled evenly throughout the domain. The middle panel of figure 5.5 shows the representation built using the q parameter only. For this representation, we have very high osculations throughout the domain: high resolution wavelets contain more than enough points in their support. Their contribution is specially important on the discontinuity and the borders of the domain.

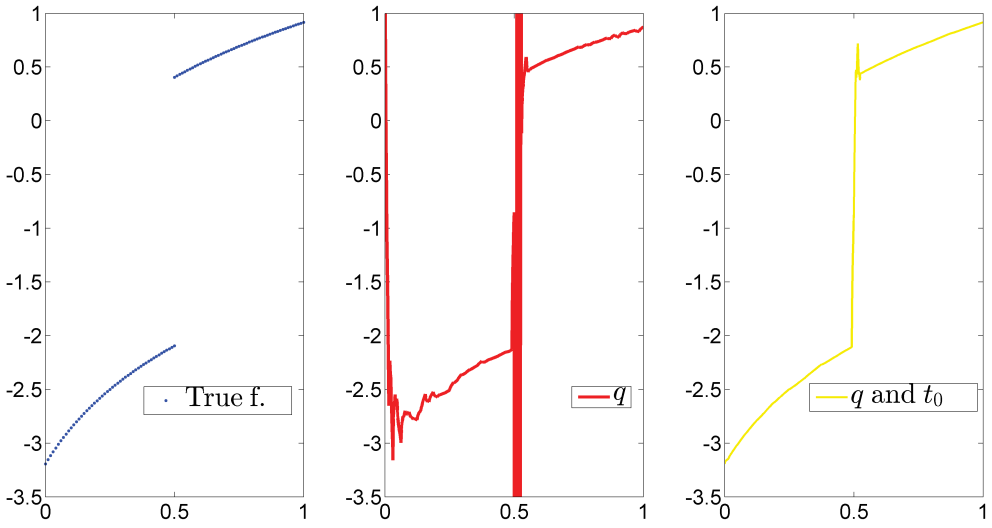


Figure 5.5: On the left, the true function. On the middle we have the prediction built using only the q parameter. On the right, the decomposition built using the wavelets selected by using the same q and t_0 as in algorithm 4 is shown.

By using the threshold on the coefficients along the same q, we obtain the plot on the right panel of figure 5.5. All of the high frequency oscillations on the borders disappeared improving the reconstruction.

We can see the wavelets selected for both decomposition on figure 5.6. There are 66 q-selected wavelets. They form a complete set up to resolution 5. This explains the high frequency oscillations throughout the domain mentioned above.

With the same q and a threshold t_0 we selected 37 wavelets. Most of the selected wavelets are located at the discontinuity, at its right or at the

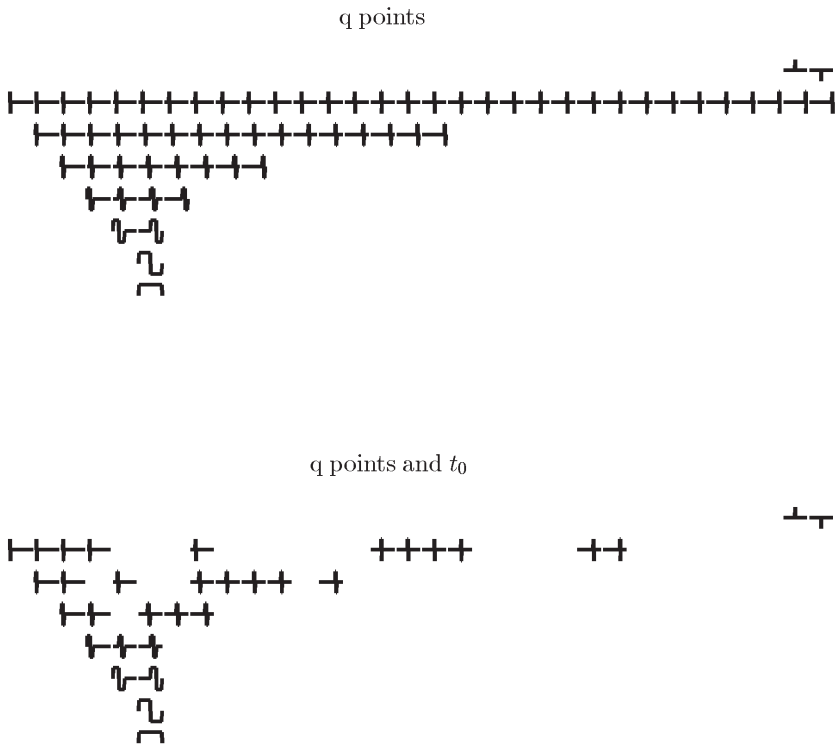


Figure 5.6: The q selected wavelets are shown in the top panel and the $q - t_0$ ones in the bottom. We use Haar wavelets to represent the wavelets used in the actual decomposition.

borders. This is reflected on the prediction: there is an oscillation right after the jump. The selected wavelets reflect the fact that there are two regular parts and a jump.

5.1.3 Some remarks on the adaptive wavelet decomposition algorithm

The first is that wavelets are usually used in a different configuration. The data should be observed over a dyadic grid and the wavelet coefficients are obtained by fixing a highest resolution and filtering them until we reach the 0^{th} resolution - see appendix B for details.

Also, most wavelets do not have an analytic formula. To evaluate them

at an arbitrary input we use an approximation scheme described in part 2 of appendix B.

In the next section we introduce the multi-fidelity version of this algorithm. We apply the adaptive wavelet decomposition to every data set and their corresponding differences.

5.2 Multi-fidelity wavelet adaptive regression

We are given two data sets $(\mathcal{X}_1, y_{1,n_1})$ and $(\mathcal{X}_2, y_{2,n_2})$ generated by two unknown functions f_1 and f_2. We assume that f_1 is an approximation of f_2 that is easier to evaluate.

The goal of the multi-fidelity algorithm is to add observations sequentially by using the information given by the selection procedure of algorithm 4. To do this we focus on two aspects of multi-fidelity regression:

1. gain extra information about f_2 by looking at the approximate data $(\mathcal{X}_1, y_{1,n_1})$;

2. learn the difference between the two unknown functions.

We propose a procedure that looks at the $q - t_0$ selected coefficients of the decomposition of f_1 and f_2 and those of their difference. We note $d_{j,k}^{(l)}$ the coefficients of the $w_{j,k}^{(l)}$ wavelet in the decomposition of f_l for $l = 1, 2$. Let $f_d = f_2 - f_1$ the difference between the two unknown functions. We note $d_{j,k}^{(d)}$ its corresponding decomposition coefficients.

We use algorithm 4 to select the wavelets on the decompositions of f_1

and f_d and write

$$f_1(x) \approx \widehat{c}^{(1)}\ \phi(x) + \sum_{j=0}^{J}\sum_{k=0}^{2^j-1} \widehat{d}^{(1)}_{j,k}\ w_{j,k}(x)$$

$$f_d(x) \approx \widehat{c}^{(d)}\ \phi(x) + \sum_{j=0}^{J}\sum_{k=0}^{2^j-1} \widehat{d}^{(d)}_{j,k}\ w_{j,k}(x).$$

Then, the decomposition of the target function is

$$f_2(x) \approx (\widehat{c}^{(1)} + \widehat{c}^{(d)})\ \phi(x) + \sum_{j=0}^{J}\sum_{k=0}^{2^j-1} (\widehat{d}^{(1)}_{j,k} + \widehat{d}^{(d)}_{j,k})\ w_{j,k}(x).$$

If $\widehat{d}^{(d)}_{j,k} = 0$, then $\widehat{d}^{(2)}_{j,k} \approx \widehat{d}^{(1)}_{j,k}$. We can interpret a zero difference coefficient as the sign of a location where f_1 and f_2 are similar. Conversely, when $\widehat{d}^{(d)}_{j,k} \neq 0$ we have the coefficient of a wavelet that makes a significant contribution on the decomposition that we can interpret as the sign of a location where the two functions are different.

By being able to distinguish the locations where the target function and the approximation are similar we can decide when to look at f_1 to have some information about f_2. This is specially convenient since we are able to evaluate f_1 more easily.

The algorithm starts with a $q - t_0$ decomposition of f_1, f_2 and f_d, initiated at resolution j_0. It is a coarse to fine procedure that starts at j_0 and looks for wavelets with locations k such that $\widehat{d}^{(1)}_{j_0,k} \neq 0$, $\widehat{d}^{(d)}_{j_0,k} = 0$ and $\widehat{d}^{(2)}_{j_0,k} = 0$. If there is such a combination of resolution and location we add p observations of f_2 on the support of $w^{(2)}_{j_0,k}$ and erase the children of $w^{(1)}_{j_0,k}$ from the search. The selection process is repeated for the next resolutions $j_0 + 1, j_0 + 2, ...$ Once we have looked in all the resolution of the first decomposition, we make a new one and start again. This is algorithm 5 below.

So, to add additional observations we run algorithm 4 on f_1, f_2 and

f_d and allocate them at the zones in which we haven't explored f_2; the difference is small and f_1 has $q - t_0$ selected wavelets. This is $\widehat{d}^{(2)}_{j_0,k} = 0$, $\widehat{d}^{(d)}_{j_0,k} = 0$ and $\widehat{d}^{(1)}_{j_0,k} \neq 0$ that is noted $K_j := K_j^{(1)} \cap \left(K_j^{(d)}\right)^c \cap \left(K_j^{(2)}\right)^c$.

Algorithm 5 Multi-fidelity adaptive wavelets sequential design.

procedure
  Fix j_0 and q.
  while Evaluation budget lasts **do**
    Apply algorithm 4 to $(\mathcal{X}_1, y_{1,n_1})$, $(\mathcal{X}_2, y_{d,n_2})$ and $(\mathcal{X}_2, y_{2,n_2})$.
    Set $j \leftarrow j_0$.
    Build $K_j := K_j^{(1)} \cap \left(K_j^{(d)}\right)^c \cap \left(K_j^{(2)}\right)^c$.
    while $K_j \neq \emptyset$ **do**
      For all $k \in K_j$
      add p points to the support of $w_{j,k}^{(2)}$ and
      erase the children of $w_{j,k}^{(1)}$ from $K_{j+1}^{(1)}$.
      Set $j \leftarrow j + 1$.
      Build $K_j := K_j^{(1)} \cap \left(K_j^{(d)}\right)^c \cap \left(K_j^{(2)}\right)^c$.
    Form the new data sets $(\mathcal{X}_2, y_{d,n_2})$ and $(\mathcal{X}_2, y_{2,n_2})$ adding the new evaluations.
  return $K_j^{(1)}$, $K_j^{(d)}$ and $K_j^{(2)}$.

The following example shows how the algorithm works. We consider two functions f_1 and f_2 made up of three parts. They are identical on the first third. On the middle, the approximation f_1 is a shifted version of the target f_2. Finally, on the right f_1 represents a coarse approximation. The two functions are shown on figure 5.7.

We sample the two functions on the same input set on the left. On the middle, where the difference is small,we sample f_2 at 1 every 10 inputs at which we sample f_2. On the third part there we sample f_1 three times more than f_2. We use Daubechies wavelets.

Figure 5.9 shows the $q - t_0$ selected wavelets of three steps of algorithm 5. We can see in panel 5.9a that the maximal resolution is 7. The first

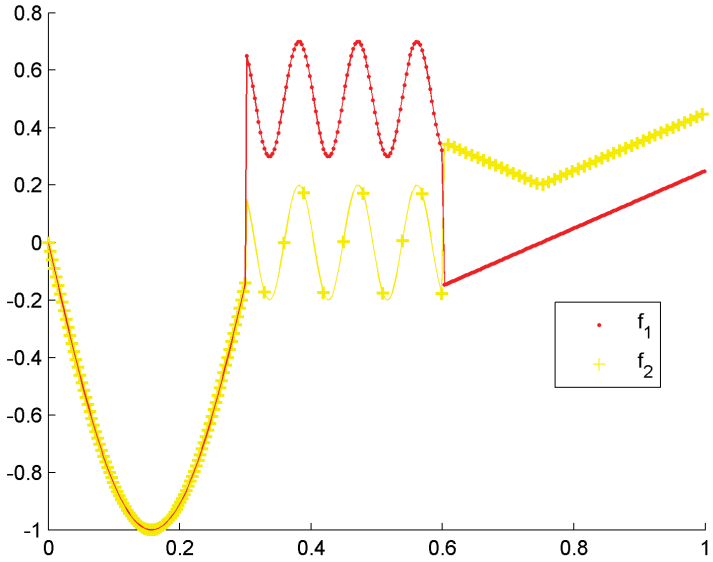


Figure 5.7: The two functions f_1 and f_2, are sampled at the dots shown in the plot. On the middle f_1 is a shifted up version of f_2. The right part of f_1 represents an approximation of f_2.

resolution at which we add points is 5. With this in mind we see that the selected wavelets on the decomposition of f_1 are the first 20 out of the 32 at resolution 5. For f_2, we select the first 11, that corresponds to the first third, and some at the end of the second third, 19 through 24. We see that the difference in these locations is zero. We add the points,and after a new $q - t_0$ selection figure 5.9b is obtained. We see that some wavelets at resolution 5 were chosen.

The next search is mainly focused on the resolution 6. We see that some wavelets at this resolution were added in step 3 in 5.9c. The reconstruction using these wavelets is shown in figure 5.8 below.

By using this type of decomposition we obtain detailed information about the data at different scales and locations. At each step, we can identify zones with potential need of further local refinement and use this information to add observations. The selected wavelets give a picture of the relationship between the three functions.

A drawback of this method is that it requires a large number of obser-

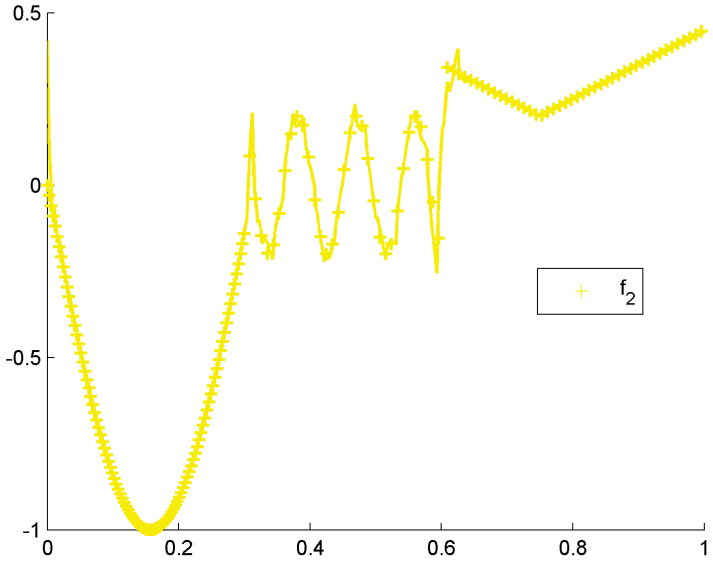


Figure 5.8: Final data set for f_2 and the predicted function on $[0, 1]$.

vations and this situation deteriorates for high dimensional inputs where wavelets are built as tensor products. When the budget for observations is small, choosing a more simple wavelet, like Haar's, can be an alternative to explore the functions.

5.3 Conclusion

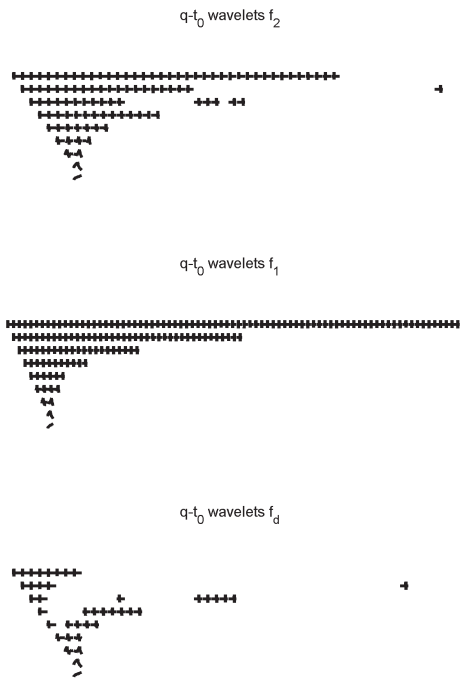
Adaptive Wavelet decomposition. In this chapter we introduce an algorithm based on a very particular decomposition that adds wavelets at higher resolutions sequentially.

The algorithm uses the number of points q and the size of the estimated wavelet coefficients. The number of points tells us where the functions are unexplored and the size of the coefficients the local smoothness.

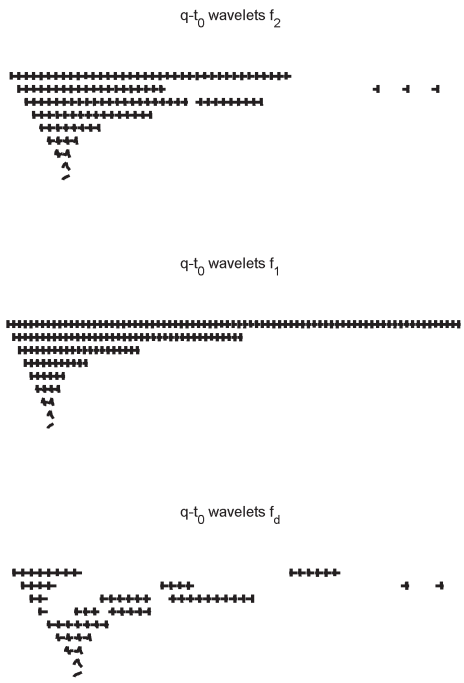
Sequential design. In the multi-fidelity part we consider two data sets generated by two unknown functions f_1 and f_2. By using the information of the adaptive wavelet decomposition we learn the difference between the two functions f_d. We use f_1 to explore zones where we have not explored

f_2 and where the difference is small. Then, we add observations in the support of the selected wavelets at a precise scale and location.

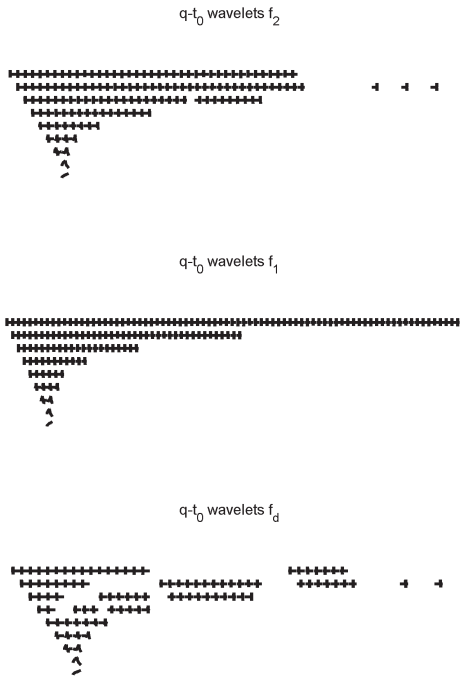
The algorithms presented in this chapter give a picture of the zones, of an unknown function, of local regularity and the zones that could be interesting to explore. By applying this ideas to multi-fidelity it is possible to localize the differences between the data sets. This tools are an alternative to the Gaussian models that needs a lot of observations and that is difficult to apply in dimensions bigger than 2. They could be used to explore and understand complicated data sets, their differences and relationship.



(a) Step 1.



(b) Step 2.



(c) Step 3.

Figure 5.9: Selected wavelets for the three functions f_1, f_d and f_2. Each panel represents a step of algorithm 5.

Chapter 6

Discussion

In this chapter we summarize the contributions of this thesis. Then, the limitations of this study are mentioned and we rise some questions related to the models studied.

6.1 Summary of the contributions

The main contribution of this thesis is to propose two new multi-fidelity regression models.

Gaussian processes model. The first one is based on modeling each data set and their differences by using Gaussian processes. We focused on studying the relationship between two successive levels of fidelity. In chapter 2 we extend the models of [30, 35] to the case of a polynomial relationship. Then, in chapter 3 we consider a model in which the relationship is unknown and estimated using locally linear polynomials. We show that by estimating the relationship we can incorporate data that would be useless otherwise.

Nested designs and EM. All of these extensions kept the sequential prediction property under the condition of nested input sets. In chapter 4 we propose an algorithm to build this nested sets by using the entropy of the

outputs and in chapter 2 we proposed an EM type algorithm to extend all of the models above to the case of disjoint input sets. Finally, we provided simple proofs that showed that the sequential prediction property is due to the independence structure of the processes involved and not the fact that we are working with Gaussian processes.

Adaptive wavelets. The second model is an adaptive multi-fidelity regression technique based on a particular wavelet decomposition proposed in [8]. In chapter 5 we propose a sequential algorithm that uses the different data sets to explore an unknown function in the zones where it varies the most. An adaptive design algorithm based on this concepts allocates observations at specific locations.

We describe and analyze a multi-fidelity case study motivated by a practical problem of importance in France: the aging of nuclear power plants. Each one of the three outputs we study is a different multi-fidelity example: for G there is a linear relationship between levels; for W the relationship is nonlinear and L is an example in which it is difficult to use the approximate data to improve the prediction.

6.2 Limitations

Estimating the relationship. The theoretical properties of the locally linear estimator of chapter 3 are not studied. It is not clear what types of relationship are possible to estimate efficiently. This is an important step that needs to be taken to find out about the possible limits of this method.

Nested designs. A more in detail study of the performance of the $Dmax$ design strategy has to be done. In some preliminary studies it gives better results in lower dimensions.

Case study. The conclusions about the influence of the parameters and their interactions should be explored further in the simulator to possibly

find a satisfactory physical explanation of the results.

The fact that the mesh size affected the influence of the parameters was not explored. This could be crucial to accelerate the study of these complex simulators.

Adaptive wavelets. This method has to be studied more extensively. We have only shown one dimensional examples in which the data is sampled from known functions. This method should also be tested in examples where the objective is not prediction but to explore complicated data sets and their relationships. This could be applied to classification.

6.3 Open questions

New locally linear estimator. The locally linear regression problem presented in chapter 3 is different from the ones found in the literature: the inputs - in this case the coarsest fidelity level - and the observation errors - this is the difference process - have a complicated covariance structure. A first step to understand this nonparametric regression problem would be to compute the asymptotic bias and variance of the estimator. It turns out that the asymptotic bias is the same as the one of the usual locally linear estimator built using iid observations. Computing the variance is the difficult part mainly because of the covariance structure of the inputs.

Relationship between two fidelity levels. Sometimes the relationship between the outputs of two successive fidelity levels is not a function. Figure 6.1 shows the relationship between the maximum pressures $p(K, n^*; 20)$ and $p(K, n^*, 40)$ of chapter 3. A possible extension of the method would be to estimate the relationship using other non parametric techniques like the nonparametric modal regression described in [10] or functional data analysis like in [46].

Order of fidelity. Throughout the thesis we assumed that we were

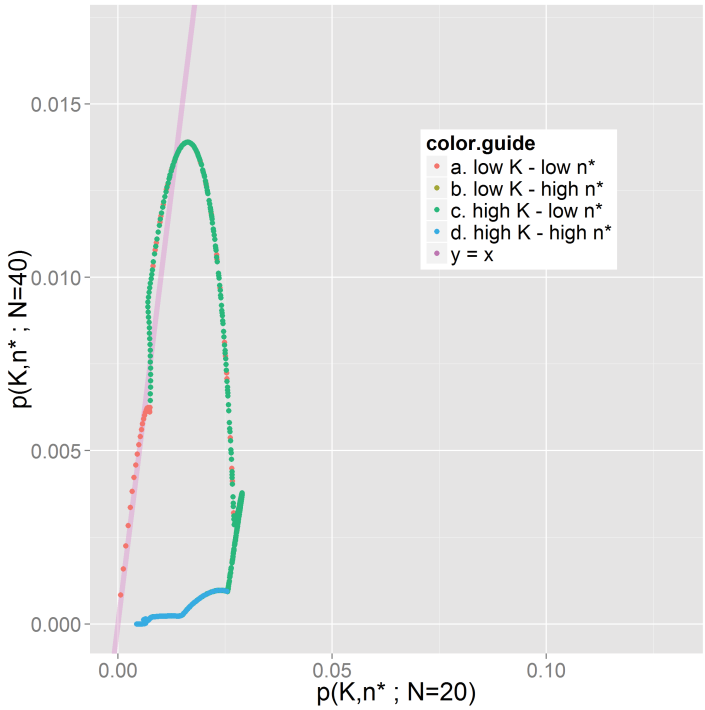


Figure 6.1: Pairs $(p(K, n^*; 20), p(K, n^*, 40))$ of the numerical example in chapter 3.

given a series of data sets ordered by their fidelity. This order can arise naturally in for example the numerical solution of a differential equation, but in general it is not clear how to choose it. For the Gaussian process case, we showed that the recursive prediction formulation is a consequence of the independence structure of the model. The problem of learning the dependence structure of a model is one of the main subjects of the Probabilistic Graphical models literature - see [31]. A Probabilistic Graphical model is a way of encoding a dependency structure by using a graph. We can choose the graph that maximizes a score. The most simple way of computing a score is to compute the log-likelihood of the data for a given graph. In our case, since we work with few data sets and we know what type of dependency we are looking for, we can search on the whole space of possible models and choose the best.

Finding the dependency structure and thus the order of fidelity can be a goal in itself. Determining which data sets are the closest can have different applications. Finally, these methods can be used to find dependency structures that are more complex than the ones we studied in this thesis.

Study of the adaptive wavelet decomposition algorithm. The original adaptive wavelet regression algorithm was proposed for B-Spline wavelets. These are a very particular type of multi-resolution analysis in which the scaling and wavelet functions are piecewise polynomials with an explicit formula. The question of whether the minimization problem to find the coefficients of the decomposition has a solution has only been answered for B-Splines. The solution depends on the fact that the observations should be well distributed and this can be easily defined for piecewise polynomials.

Also how to choose the parameters of this particular algorithm is also an open problem even for the one data set case. For the number of points q we could use a cross validation approach in which we consider a risk function with a penalty term on the number of points in the support of the wavelets - something similar is done to prune a tree grown by methods like CART or MARS [27]. This is something we cannot do to fix the coefficient size parameter t_0. For this parameters we can try several different values and choose the best result.

6.4 Conclusion

Using approximate information about an output to improve its prediction is an interesting and fairly unexplored subject. Multi-fidelity regression should be expanded and applied to more practical cases. However, determining which kind of data can be incorporated to enhance the prediction of an objective remains an open problem. I hope that the techniques explored in this work will help in the understanding of this question.

Appendix A

Gaussian processes and Bayesian estimation

A.1 Definitions

We define Gaussian vector as follows.

Definition 1. *A Gaussian vector is a vector of random variables such that any linear combination of its coordinates has a Gaussian distribution.*

When we work with a Gaussian vector $Y_n := [Y_{x_1}, ..., Y_{x_n}]^T$, we note EY_n the vector with means in each coordinate $[EY_{x_1}, ..., EY_{x_n}]^T$ and $\Gamma(n)$ its covariance matrix. If $|\Gamma(n)| := \det(\Gamma(n)) \neq 0$, we note $\Lambda(n)$ the inverse $\Gamma(n)^{-1}$. The density function of the vector at y_n is

$$f_{Y_n}(y_n) = \frac{1}{(2\pi)^{n/2}|\Gamma(n)|^{1/2}} \exp\left(-\frac{1}{2}(y_n - EY_n)^T \Lambda(n)(y_n - EY_n)\right).$$

A Gaussian process is defined using Gaussian vectors as follows.

Definition 2. *A Guassian process is a random process $\{Y_x : x \in \mathcal{X}\}$ such that for any finite n and any set of indexes $\{x_1, ..., x_n\}$ in $\mathcal{X}$, the vector $[Y_{x_1}, ..., Y_{x_n}]^T$ is a Gaussian vector.*

A Gaussian process is entirely defined by the mean function $x \longmapsto EY_x$ and the covariance function $(x, x') \longmapsto Cov(Y_x, Y_{x'})$.

In the manuscript we consider covariances that depend only on the indexes so we write $Cov(Y_x, Y_{x'}) = \Gamma(x, x')$. Given a vector $Y_n := (Y_{x_1}, ..., Y_{x_n})^T$ we note $\Gamma(n, x)$ the vector of covariances with entries $\Gamma(x_i, x)$, $i \in \{1, ..., n\}$.

A.2 Bayesian estimation for Gaussian processes

Here we assume a parametric form for the mean and covariance functions of a Gaussian process. We give the formula for the prediction error for the case in which the mean parameters are unknown.

Before writing the explicit formula for the prediction error we are looking for we prove a very useful proposition taken from the book Pattern Recognition and Machine Learning by Christopher Bishop [5].

Proposition 4. *We consider two Gaussian random vectors defined as*

$$X \sim N(\mu_0, \Gamma_0)$$
$$Y|\{X = x\} \sim N(Ax + b, \Gamma).$$

Then,

$$Y \sim N(A\mu_0 + b, \Gamma + A\Gamma_0 A^T).$$

Proof. We note Λ_0 and Λ the matrices Γ_0^{-1} and Γ^{-1}.

The way you prove this is by first figuring out what is the joint distribution of X and Y. We note $Z := [X, Y]^T$, $z := [x, y]^T$ and write

$$f_Z(z) = f_{Y|X}(y|x) f_X(x).$$

You can check that the product of the density functions of two Gaussian random variables is Gaussian. Then, Z has a Gaussian distribution.

With this in mind we take the logarithm of $f_Z(z)$ to get

$$\begin{aligned}\log f_Z(z) &= \log f_{Y|X}(y|x) + \log f_X(x) \\ &= -\frac{1}{2}(y-(Ax+b))^T\Lambda(y-(Ax+b)) \\ &- \frac{1}{2}(x-\mu_0)^T\Lambda_0(x-\mu_0) + rest.\end{aligned}$$

We are only interested in the first two terms. To identify the covariance matrix of Z we look at the quadratic terms in x and y. They are

$$-\frac{1}{2}(y^T\Lambda y - y^T\Lambda Ax - x^TA^T\Lambda y - x^TA^T\Lambda Ax) - \frac{1}{2}(x^T\Lambda_0 x).$$

We rewrite them using a matrix notation to get

$$-\frac{1}{2}\begin{bmatrix}x\\y\end{bmatrix}^T \underbrace{\begin{bmatrix}A^T\Lambda A+\Lambda_0 & -\Lambda A\\ -A^T\Lambda & \Lambda\end{bmatrix}}_{\mathrm{R}}\begin{bmatrix}x\\y\end{bmatrix}.$$

The covariance matrix of Z is R^{-1}. By using the block wise inversion formula we can write R^{-1} as

$$\begin{bmatrix}\Lambda_0^{-1} & \Lambda_0^{-1}A^T\\ A\Lambda_0^{-1} & \Lambda^{-1}+A\Lambda_0^{-1}A^T\end{bmatrix}.$$

To get the formula for the mean of Z we look at terms that look like $\frac{1}{2}Z^TRE[Z]$, this is linear forms with x or y on the left. They are

$$\frac{1}{2}(y^T\Lambda b - x^TA^T\Lambda b) + \frac{1}{2}(x^T\Lambda_0\mu_0).$$

In matrix form

$$\frac{1}{2}\begin{bmatrix}x\\y\end{bmatrix}^T\begin{bmatrix}-A^T\Lambda b+\Lambda_0\mu_0\\ \Gamma b\end{bmatrix}.$$

From this expression you can figure out that

$$EZ = R^{-1} \begin{bmatrix} -A^T \Lambda b + \Lambda_0 \mu_0 \\ \Lambda b \end{bmatrix}$$
$$= \begin{bmatrix} \mu_0 \\ A\mu_0 + b \end{bmatrix}.$$

Finally, we have the whole distribution for Z. It is

$$\begin{bmatrix} X \\ Y \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_0 \\ A\mu_0 + b \end{bmatrix}, \begin{bmatrix} \Lambda_0^{-1} & \Lambda_0^{-1} A^T \\ A\Lambda_0^{-1} & \Lambda^{-1} + A\Lambda_0^{-1} A^T \end{bmatrix} \right).$$

This gives in particular the distribution of Y. □

We consider a Gaussian process $\{Y_x : x \in \mathcal{X}\}$. We assume that its mean function can be written as $f(x)^T \mu$ where $f(x)$ is a vector of known functions and μ is a vector of unknown parameters. The covariance $Cov(Y_x, Y_{x'})$ or $\Gamma(x, x')$ is defined as $\sigma^2 \gamma(x, x'; \theta)$ for a vector of parameters θ and a parameter σ^2.

To have a more simple notation we assume that σ^2 and θ are known. We know the distribution of $Y_x | \{Y_n = y_n, M = \mu\}$ and we want to compute that of $Y_x | \{Y_n = y_n\}$.

To do this we first assign a prior normal distribution to M and we compute the posterior distribution of $M | \{Y_n = y_n\}$. Since

$$f_{M|Y_n}(\mu | y_n) \propto f_{Y_n|M}(y_n | \mu) f_M(\mu)$$

we are interested in the following distributions

$$M \sim N(\mu_0, \Gamma_0)$$
$$Y_n | \{M = \mu\} \sim N(F(n)^T \mu, \Gamma(n)).$$

Where we set the distribution of M to be Gaussian and $F(n)$ is a matrix with columns $f(x_j)$ for j between 1 and n. Now we can expand the product

of distributions $f_{Y_n|M}(y_n|\mu)f_M(\mu)$. We look at the exponents

$$-\frac{1}{2}(y_n - F(n)^T\mu)^T\Lambda(n)(y_n - F(n)^T\mu) - \frac{1}{2}(\mu - \mu_0)^T\Lambda_0(\mu - \mu_0)$$

In the quadratic terms we identify the covariance matrix

$$\Gamma_{M|Y_n} := (F(n)\Lambda(n)F(n)^T + \Lambda_0)^{-1}$$

and we look at the linear terms to find the mean

$$\mu_{M|Y_n} := \Gamma_{M|Y_n}(F(n)\Gamma(n)y_n + \Lambda_0\mu_0).$$

To find the predictive distribution of Y_x given $\{Y_n = y_n\}$ we begin by working with

$$Y_x|\{Y_n = y_n, M = \mu\} \sim N(\widehat{Y}_x, \widehat{\Gamma}(x, x)).$$

First we rewrite the mean to match the distributions in the proposition.

$$\begin{aligned}\widehat{Y}_x &= f(x)^T\mu + \Gamma(n, x)^T\Lambda(n)(y_n - F(n)^T\mu) \\ &= \left(f(x)^T - \Gamma(n, x)^T\Lambda(n)^{-1}F(n)^T\right)\mu + \Gamma(n, x)^T\Lambda(n)y_n \\ &= A\mu + b.\end{aligned}$$

Then, Y_x given $\{Y_n = y_n\}$ is

$$N(A\mu_{M|Y_n} + b, \widehat{\Gamma}(x, x') + A\Gamma_{M|Y_n}A^T).$$

This is a Bayesian description of the distribution when the mean parameter is unknown and we use a Gaussian prior.

A.2.1 Some prior distributions and their posteriors

Below we list some popular choices for the prior distributions for GP regression. We write a non informative and a conjugate prior for the inference on M and (M, Σ^2).

For the mean parameter

$$M \sim 1$$
$$M \sim N(\mu_0, \sigma_0^2)$$

For the joint distribution of the mean and variance parameters the non informative prior is

$$(M, \Sigma^2) \sim 1/\Sigma$$

and the conjugate priors are

$$[M|\Sigma^2 = \sigma^2] \sim N(\mu_0, \sigma^2)$$
$$[1/\Sigma^2] \sim Gamma(a = v_0/2, \lambda = S_0/2).$$

In this case the non informative priors are not pdfs because they do not integrate to 1.

Posterior distributions

The posterior distributions for the mean are

$$[M|Y_n] \sim N(\widehat{\mu}_{MLE}(\theta), \sigma^2(\mathbf{1}^T\lambda(n;\theta)\mathbf{1})^{-1})$$
$$[M|Y_n] \sim N(\mu_{M|Y_n}, \Gamma_{M|Y_n}).$$

Where $\mu_{M|Y_n}$ and $\Gamma_{M|Y_n}$ were computed in the section above and we replace F with $\mathbf{1}$.

The variance for the mean posterior is the variance of $\mu_{MLE}(\theta)$. The mean of the Gamma posterior is $1/\widehat{\sigma}^2_{MLE}$ and variance

For the non informative priors on the mean and variance parameters we

have

$$[M, \Sigma^2|Y_n] \sim N(\widehat{\mu}_{MLE}(\theta), \sigma^2(\mathbf{1}^T\lambda(n;\theta)\mathbf{1})^{-1})$$
$$[1/\Sigma^2|Y_n] \sim Gamma\left(\frac{n-1}{2}, \frac{2}{(n-1)\widehat{\sigma}^2_{MLE}(\theta)}\right).$$

and for the conjugate priors

$$[M|\Sigma^2 = \sigma^2, Y_n] \sim N\left(\frac{\mu_0 + n\bar{Y_n}}{n+1}, (n+1)\sigma^2\right)$$
$$[1/\Sigma^2|Y_n] \sim Gamma\left(\frac{v_0+n}{2}, \frac{S_0 + S + n(\mu_0 - \bar{Y_n})^2}{n+1}\right).$$

Predictive distributions

For the inference on the mean we use the proposition at the beginning of the appendix. We get

$$[\widehat{Y_x}|Y_n] \sim N\left(A\mu_{MLE}(\theta) + b, \widehat{\Gamma}(x,x') + \sigma^2 A(\mathbf{1}^T\lambda(n;\theta)\mathbf{1})^{-1}A^T\right) \quad (A.1)$$

for the non informative prior. We already derived the distribution for the conjugate prior. It is

$$[\widehat{Y_x}|Y_n] \sim N(A\mu_{M|Y_n} + b, \widehat{\Gamma}(x,x') + A\Gamma_{M|Y_n}A^T).$$

The other two predictive distribution that can be derived are the one related to the priors on (M, Σ^2). They are a multivariate Stundent-t distribution whose mean is the same as first the one in A.2.1 but with a different variances.

Appendix B

A short introduction to Wavelets

B.1 Building wavelets and the refinement equation

Here we use filters to understand how wavelets are built. The main refferences for this part are the courses of Linear Algebra and Computational Science and Engineering I by Gilbert Strang [51], [52] as well as the article Orthonormal Bases of Compactly Supported Wavelets by Ingrid Daubechies [14].

B.1.1 Multiresolution Analysis

We define a way of looking at a function f in $L^2(\mathbf{R})$. The idea is to write f as the limit of successive approximations. Each approximation defines a resolution and hence the name.

Multiresolution starts with a family of nested spaces of functions.

$$\underbrace{V_0}_{Coarse} \subset \underbrace{V_1}_{finer} \subset \underbrace{V_2}_{even\ finer} \subset \cdots$$

This succession is such that $\bigcup_j V_j$ is dense in $L^2(\mathbf{R})$. To build them we start with V_0. It is defined as the space spanned by all the translates $k \in \mathbf{Z}$

of a unique function $\phi(x-k)$ called the scaling function. In particular we require that for any $\phi \in L^2(\mathbf{R})$ such that $\phi \in V_0$, then for all $k \in \mathbf{Z}$, $\phi(\cdot - k) \in V_0$. The projection of f into V_0 represents a coarse approximation or, in other words, V_0 represents the coarsest resolution.

The next space contains all the linear combinations of $\phi(2x-k)$ for $k \in \mathbf{Z}$ and j-space V_j is spanned by all the $\phi_{j,k}(x) := \phi(2^j x - k)$. For this to make sense we need that $\phi_{j,k} \in V_j$ if and only if $\phi_{j+1,k} \in V_{j+1}$.

A multiresolution analysis is the succession of subspaces V_j that satisfy the properties above. Namely,

(1) $\bigcup_j V_j$ is dense in $L^2(\mathbf{R})$.

(2) Any $\phi \in L^2(\mathbf{R})$ that is also in V_0 is such that for all $k \in \mathbf{Z}$, $\phi_{0,k} \in V_0$.

(3) $\phi_{j,k} \in V_j$ if and only if $\phi_{j+1,k} \in V_{j+1}$.

(4) There exists a function ϕ such that its translates are a basis for V_0.

For the first example, we build the succession of spaces V_j by defining $\phi(x)$ as equal to 1 in $[0,1]$ and 0 elsewhere. Then, V_0 is the set of piece-wise constant functions on unit intervals and V_1 has piece-wise constant functions on half intervals. In this example the idea of resolution becomes clear: As j increases we get a better idea of f up to resolution 2^{-j}, the size of the support of $\phi_{j,k}(x) = \phi(2^j x - k)$. Figure B.1.1 shows the reconstruction of a function in V_j on top and one in V_{j+1} at the bottom.

This construction has several good properties that we would like to keep like the fact that all the translates of the scaling function are orthogonal and compactly supported. But one big drawback is that this decomposition is very poor at approximating smooth functions.

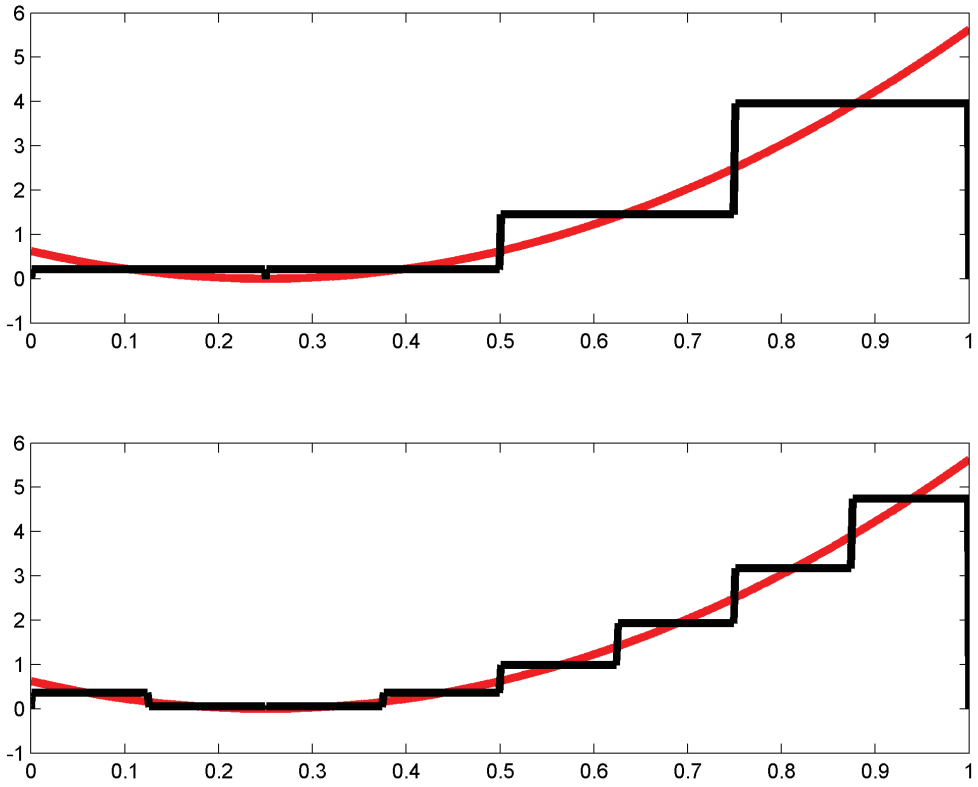


Figure B.1: Top: reconstruction of the red curve using constant piece wise functions on intervals twice as big as the ones used on the bottom figure.

B.1.2 The refinement equation

We need different multiresolution spaces when f is smooth. The key idea to understand how to build a new multiresolution succession is given by the fact that V_1 contains V_0. This means that in particular we can write that

$$\phi(x) = 2 \sum_{k=-\infty}^{\infty} h_0(k)\phi(2x-k)$$

for some coefficients $h_0(k)$. This relationship is called the scaling equation. It is the discrete convolution of h_0 with $\phi(2x)$. We are filtering when we pass from one resolution to next. In order to understand what we would like to filter lets go back to ϕ equal to 1 on $[0,1]$ and zero everywhere else. For this example you can see that $\phi(x) = \phi(2x) + \phi(2x-1)$ so that $h_0(0)$ and $h_0(1)$

are 0.5 and the rest of the entries are 0. This $h_0 = (...0, 0.5, 0.5, 0, ...)$ defines a low pass filter. From now on we will only write the nonzero coefficients of the filter for short. With this notation $h_0 = (...0, 0.5, 0.5, 0, ...) =: (0.5, 0.5)$.

What type of filter is h_0? If h_0 is a low pass filter then it should leave unchanged low frequency oscillating signals like the constant signal $x_k = 1$ for all $k \in \mathbf{Z}$. This is true: the resulting signal is $y = (x * h_0)$ where $y_k = \sum_{l=-\infty}^{\infty} h_0(l)x_{k-l} = 0.5 \cdot x_k + 0.5 \cdot x_{k-1} = 1$. This filter also sets to 0 fast oscillating signals like $x_k = (-1)^k$ since $y_k = 0.5 \cdot x_k + 0.5 \cdot x_{k-1}$ where two successive terms of x_k have alternating signs.

To test all the remaining frequencies we compute the convolution with $x_k = e^{isk}$ where s, the frequency, is between $-\pi$ and π. The resulting signal has entries $y_k = 0.5 \cdot e^{isk} + 0.5 \cdot e^{is(k-1)}$ that we can write as $0.5 \cdot e^{isk}(1 + e^{-is})$. We note $H_0(s) = 0.5 \cdot (1 + e^{-is})$. When we input a pure frequency e^{isk} we get $H_0(s)e^{isk}$. We plot the magnitude of this function in figure B.2.

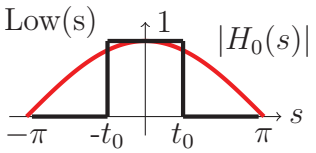


Figure B.2: The magnitude $|H_0(s)|$ is plotted in red. The black function is an ideal low pass filter that cuts the signal at $\pm t_0$.

So in order to build the succession of V_j's we use a low pass filter successively. Notice that this low pass filter is far from ideal. We would like to have a plot shaped as a box like $Low(s)$.

When we pass from one resolution to another we filter the high frequencies. Wavelets will capture the high frequencies. They are define by a high pass filter that we note h_1. For our running example the low pass filter was defined by the averaging $y_k = 0.5 \cdot x_k + 0.5 \cdot x_{k-1}$, the high pass filter is given by the difference $y_k = 0.5 \cdot x_k - 0.5 \cdot x_{k-1}$. The coefficients are $h_1(0) = 0.5$, $h_1(1) = -0.5$ and 0 elsewhere. $H_1(s)$ is equal to $0.5 \cdot (1 - e^{-is})$ and looks like the red curve below.

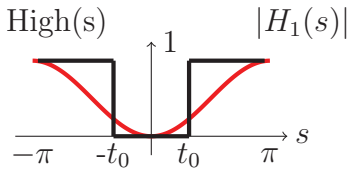


Figure B.3: The magnitude $|H_1(s)|$ is plotted in red. The black function is an ideal high pass filter that cuts the signal inside $[-t_0, t_0]$.

We build wavelets w by using the following scaling equation

$$w(x) = 2 \sum_{k=-\infty}^{\infty} h_1(k)\phi(2x - k).$$

We have $w(x) = 2(0.5 \cdot \phi(2x) - 0.5 \cdot \phi(2x - 1))$ for ϕ equal to the box function. In this case, w is a function that is 1 on the first half interval and -1 on the second. It looks like a local wiggle as shown on figure B.4. In general we choose $h_1(k)$ as $(-1)^k h_0(k)$. This is sufficient to assure that

Figure B.4: Haar's scaling and wavelet functions, ϕ and w.

the translates of the resulting wavelets constitute a basis of the orthogonal space of V_0 in V_1, see [14]. This is Haar's multiresolution analysis. Finally, if we note W_0 the space of wavelets, we have that $V_1 = V_0 \oplus W_0$. And if we push this idea to the limit, we have $L^2(\mathbf{R}) = V_0 \oplus W_0 \oplus W_1 \oplus \cdots$. The function f can be written as a linear combination of the basis of V_0 plus linear combination of the bases of the W_j's like

$$f(x) = \sum_{k=-\infty}^{\infty} c_k \ \phi(x - k) + \sum_{j=0}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k} \ w(2^j x - k).$$

In the general case we use the same decomposition as the one above but we use a different h_0 and a different h_1. As we discussed before we can decide the form of the filter by looking at the Fourier transforms of

h_0 and h_1. Once they are fixed we try to determine whether the scaling equation has a solution. If it does, we look at the properties of the spaces we obtain. In the next section we will see that in order to understand a multiresolution analysis succession we only need to look at the filters h_0 and h_1.

B.2 w(x) for every x

To define a scaling function and a wavelet, we need to select two filters, h_0 and h_1. We argued that picking $h_0 = (0.5, 0.5)$ and $h_1 = (0.5, -0.5)$ was not suited to represent smooth functions. In this section we will look at other filters and we will explain why is it difficult to compute $\phi(x)$ and $w(x)$ at any given point.

B.2.1 Filters or functions?

An interesting example begins with the sequence $h_0 = (0.25, 0.5, 0.25)$. Applying this type of filter is the same as averaging. If you look closely to h_0 you can see that this filter is a centered average: $0.25(x_{k+1} + 2x_k + x_{k-1})$. As expected this gives rise to a more regular scaling function, the hat function $\wedge$.

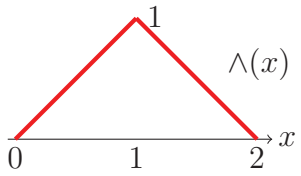


Figure B.5: Convolution of two Haar scaling functions.

The function $\wedge$ and h_0 are related in many ways. First, the regularity. Then, it is also interesting that the two have the same support length, 3. Furthermore, the convolution $(h_0 * h_0) = (1, 4, 6, 4, 1)/16$ gives $(\wedge * \wedge)$ which are cubic splines, as a scaling function. It is fair to ask whether it is

necessary to define ϕ and w explicitly or if having h_0 and h_1 is enough to make a multiresolution analysis.

It turns out that many properties of a multiresolution analysis depend on the filters and the most important examples of wavelets are not defined explicitly as it is easier to work with h_0 and h_1 directly. For example one of the Daubechies filters has non zero entries proportional to $1+\sqrt{3}, 3+\sqrt{3}, 3-\sqrt{3}, 1-\sqrt{3}$. Computing $w(x)$ for any x requires some additional work. We look a the conditions needed to solve the scaling equation.

In [15] I. Daubechies studies the conditions on h_0 under which the scaling equation has a solution. To understand the conditions we compute the Fourier transform of the scaling equation:

$$\begin{aligned}\mathcal{F}(\phi)(s) &= 2\sum_{k=-\infty}^{\infty} h_0(k)\int \phi(2x-k)e^{-ixs}dx \\ &= \left(\frac{1}{2}\sum_{k=-\infty}^{\infty} 2h_0(k)e^{-ik\frac{s}{2}}\right)\left(\int \phi(u)e^{-iu\frac{s}{2}}du\right) \\ &= P\left(\frac{s}{2}\right)\mathcal{F}(\phi)\left(\frac{s}{2}\right)\end{aligned}$$

By repeatedly using the scaling equation on φ we have that

$$\mathcal{F}(\phi)(s) = P\left(\frac{s}{2}\right)P\left(\frac{s}{2^2}\right)\cdots P\left(\frac{s}{2^j}\right)\mathcal{F}(\phi)\left(\frac{s}{2^j}\right).$$

The conditions for the existence of a solution of the scaling equation depends on the limits

$$\begin{aligned}\lim_{j\to\infty} P\left(\frac{s}{2^j}\right) &= P(0) = \sum_{k=-\infty}^{\infty} h_0(k) \\ \lim_{j\to\infty} \mathcal{F}(\phi)\left(\frac{s}{2^j}\right) &= \mathcal{F}(\phi)(0) = \int \phi(x)dx\end{aligned}$$

Theorem 2.1 of [15] states that if $P(0) = 1$ then, there is a non-trivial

solution to the scaling equation. Furthermore, $\mathcal{F}(\phi)(0) \neq 0$ and ϕ has a compact support (in fact it has the same support as h_0 see lecture 6 of [2]).

We can see that in order to build a multiresolution analysis framework we can work directly with h_0 and h_1.

A more precise description of the relationship between h_0 and ϕ can be found in the course Time-Frequency and Wavelet Analysis by Selin Aviyente [2] and Two-Scale Difference Equations I and II by Ingrid Daubechies and Jeffrey Lagarias [15] and [16].

B.2.2 Iterative approximations for $\phi(x)$ and $w(x)$.

We need to evaluate $\phi(x)$ and $w(x)$ at any x. All the results below can be found in [15] and [16] and in a report by Brani Vidakovic [7].

Here we consider that we are given a scaling equation that can be solved and whose solution ϕ is compactly supported. If the values of $\phi(n)$ are known for all $n \in \mathbf{Z}$, then it is possible to compute the values of $\phi(n/2^j)$ for any $j \in \mathbf{N}$, by using a recursive procedure.

We use the scaling equation to define the linear functional $V\phi$ described bellow.

$$V\varphi(x) = \sum_{k=-\infty}^{\infty} h_0(k)\phi(2x-k).$$

We note $\varphi_{j+1} = V\varphi_j$ where the initial function φ_0 is fixed in advance. The choice of φ_0 will link this equation to spline theory.

If you set ϕ_0 as an interpolating linear spline with knots $n \in \mathbf{Z}$, ϕ_j is a linear spline with knots on points of the form $n/2^j$. The values of $\phi_j(n/2^j)$ are the values of ϕ at the same points. Finally, ϕ_j converges to ϕ. If ϕ_0 is a spline with m continuous differentials then the differentials, up to order m, of ϕ_j converge to those of ϕ. This results are given in theorems 4.1 and

4.2 of [15].

To evaluate ϕ at any point we look at the functional V and use theorem 2.2 of [16]. Here we do not need to know the values of ϕ at any point: We apply V successively but the initial entry is given by the scaling equation directly.

The initial entry for the recursive approximation is given by a matrix M. For a filter with N non zero entries it is defined as the $(N-1)\times(N-1)$ matrix with entries $h_0(2i-j)$.

$$M = \begin{bmatrix} h_0(1) & h_0(0) & 0 & 0 & \cdots & 0 \\ h_0(3) & h_0(2) & h_0(1) & h_0(0) & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & h_0(N) & h_0(N-1) \end{bmatrix}.$$

The initial entry in the recursive approximation is defined by the eigenvector a with eigenvalue 1 of M. It is the $N-1$-vector with coordinates $a_{n-1}(1-x)+a_n x$ for $n=1,...,N-1$. We define v_0 by adding $a_0 = a_N = 0$ to this eigenvector.

We apply V to v_0 successively. This can be encoded in a convenient way by using two matrices T_0 and T_1 and the dyadic expansion of x. The two matrices have one more row and column than M. In fact, it contains M. The entries of T_0 are $h_0(2i-j-1)$ and those of T_1 are $h_0(2i-j)$ with $i,j \in \{1,...,N\}$.

If we note $d_1(x), d_2(x), ...$ the coefficients of the dyadic expansion of x, theorem 2.2 of [16] implies that the infinite product of matrices satisfies

$$T_{d_1} \cdot T_{d_2} \cdots T_{d_n} \cdots = \begin{bmatrix} \phi(x) & \cdots & \phi(x) \\ \phi(x+1) & \cdots & \phi(x+1) \\ \cdots & \cdots & \cdots \\ \phi(x+N-1) & \cdots & \phi(x+N-1) \end{bmatrix}.$$

if $\max_{d_j=0,1 \ \ j=0,...,m} \|T_{d_1} \cdot T_{d_2} \cdots T_{d_m}|_{E_1}\| \leq C\lambda^m$ for some $\lambda < 1$. Where E_1 is the orthogonal space to the left eigenvalue $(1, ..., 1)$ of T_0. Also if we note $\varphi_j(x)$ the j^{th} iteration of V, then we have that it converges to the solution of the scaling equation and $\|\phi_j - \phi\| \leq C2^{-j|\ln \lambda^m|/\ln 2}$.

B.3 Filter bank

We can build a multiresolution succession from two filters. Now, we compute the multiresolution decomposition of a function f provided we know the values of f over some discrete grid of inputs.

In the next section we explain the filter bank algorithm introduced by Stéphane Mallat in A Theory for Multiresolution Signal Decomposition: The Wavelet Representation [38].

B.3.1 The filter bank algorithm

Given a function f we want to compute the coefficients of its multiresolution decomposition. The filter bank algorithm consist in taking a series of averages and differences by using the filters used to build the multiresolution analysis.

We suppose that we know f over a dyadic grid of 2^J points. For example consider that f is equal to $y = (9, 7, 4, 8)$. Then, $J = 2$ represents the maximal resolution. Suppose that we use $h_0 = (0.5, 0.5)$ and $h_1 = (0.5, -0.5)$. Then, to compute the resolution 1 representation of y we write $(0.5 \cdot 9 + 0.5 \cdot 7, 0.5 \cdot 4 + 0.5 \cdot 8)$ which is $(8, 6)$. The difference is given by applying h_1 to y. At resolution 1 we have that the difference is $(1, -2)$.

We continue to take averages and differences until we reach resolution 0. In the example, we have that at resolution 0 we get 7 for the average and 1 for the difference. Finally, since $V_2 = V_0 \oplus W_0 \oplus W_1$ we can write

that $y = (7, 1, 1, -2)$.

How is it that the outcome of the algorithm in the last paragraph is a signal when we were looking for the coefficients of the decomposition? What we actually do is we assume that the data we are given comes from a function at some maximal resolution J. Then, we take averages according to the filter we choose. To compute the difference between two successive resolutions we use h_1. This means that we are applying the filter directly to the coefficients.

More precisely, let f be a function in V_J. Then, since $V_J = V_{J-1} \oplus W_{J-1}$ we can write f as the sum of the projections into each of the two spaces. We note P_{J-1} and Q_{J-1} the projections into V_{J-1} and W_{J-1}. Then $f = P_{J-1}f + Q_{J-1}f$. We can write each of the projections using their corresponding basis expansion

$$P_{J-1}f = \sum_k c_{J-1,k}\ \phi_{J-1,k}$$
$$Q_{J-1}f = \sum_k d_{J-1,k}\ w_{J-1,k}.$$

where $\phi_{j,k}(x) = \phi(2^j x - k)$ and similarly for $w_{j,k}$.

Now we focus on $P_{J-1}f$. If the $\varphi_{J-1,k}$'s are orthonormal, then $c_{J-1,k} =< \phi_{J-1,k}, P_{J-1}f >_{L^2}$. This expression is the same as $\sum_l c_{J,l} < \phi_{J-1,k}, \phi_{J,l} >_{L^2}$ since $P_{J-1}f$ is in V_{J-1}. Finally we have that

$$\begin{aligned} < \varphi_{J-1,k}, \phi_{J,l} >_{L^2} &= \int \phi_{J,k}(0.5 \cdot x - k)\varphi_{J,l}(x - l)dx \\ &= \int \phi_{J,k}(0.5 \cdot u)\phi_{J,l}(u - (l - 2k))dx \\ &= h_0(l - 2k). \end{aligned}$$

In conclusion, we can write the coefficients at resolution $J-1$ by using the

formula

$$c_{J-1,k} = \sum_l c_{J,l}\ h_0(l-2k).$$

A similar argument can be applied to the wavelets coefficients. This is how we computed the coefficients above.

By going back to example we notice that we can go from the multiresolution analysis representation to the original vector by doing the inverse operations. The differences between two successive resolutions are smaller than the original entries of y. Removing one difference entry introduces small errors on the reconstruction of y.

This algorithm does not contain any complicated or costly operation, we only average and take differences: it is fast. On the other hand, we need to know the function over a dyadic grid, and the size of the grid determines the maximal resolution of the decomposition. Since we start at this maximal resolution and we average or difference our way back to the resolution 0 sometimes this is called a Fine to Coarse algorithm.

For a very good overview of the filter bank and wavelets the reader can consult Wavelets for Computer Graphics by Eric Stollnitz, Tony Derose and David Salesin [50].

Bibliography

[1] Arlot, S. and Celisse, A. A survey of cross-validation procedures for model selection Statist. Surv. Volume 4, pg. 1-274, 2010.

[2] Aviyente S., ECE 802-603, Time-Frequency and Wavelet Analysis, Spring 2010, http://www.egr.msu.edu/ aviyente/ece802-10

[3] Bachoc, F. Estimation paramétrique de la fonction de covariance dans le modèle de Krigeage par processus Gaussiens. Application à la quantification des incertitudes en simulation numérique Université Paris-Diderot - Paris VII, 2013.

[4] Bardina, J.E., Huang, P.G., Coakley, T.J., Turbulence Modeling Validation, Testing, and Development, NASA Technical Memorandum 110446., 1997.

[5] Bishop, C. Pattern Recognition and Machine Learning (Information Science and Statistics) Springer-Verlag New York, Inc. Secaucus, NJ, USA 2006.

[6] Bowman, A. W. and Azzalini A. Applied Smoothing Techniques for Data Analysis Published by Oxford University Press, USA, 1997.

[7] Brani, Vidakovic, Daubechies-Lagarias Algorithm in Matlab, http://gtwavelet.bme.gatech.edu/

[8] Daniel, Castaño. Adaptive Scattered Data Fitting with Tensor Product Spline–Wavelets PhD DissertationUniversität Bonn; 2005.

[9] Castaño, D. and Kunoth, A. Robust regression of scattered data with adaptive spline-wavelets.IEEE transactions on image processing Jun, 6, p. 1621–32, 2006.

[10] Chen Y.-C., Genovese C., Tibshirani R., and Wasserman L. Nonparametric Modal Regression. 2014.

[11] Coste, P. A Large Interface Model for two-phase CFD Nuclear Engineering and Design Vol. 255, pg. 38–50, 2013.

[12] Coste P., Pouvreau, J., Lavieville, J. and Boucker, M. Status of a two-phase CFD approach to the PTS issue. In XCFD4NRS Workshop, Grenoble, France. 2008.

[13] Damblin G., Couplet M., and Iooss B. Numerical studies of space filling designs : optimization algorithms and subprojection properties, Journal of Simulation. 2013

[14] Daubechies, I. Orthonormal bases of compactly supported wavelets, Comm. Pure and Appl. Math., 41 (7), pg. 909-996, 1988.

[15] Daubechies, I. and Lagarias J.C. Two-scale difference equations. I. Existence and global regularity of solutions, SIAM J. Math. Anal., 22 (5), pg. 1388-1410, 1991.

[16] Daubechies, I. and Lagarias J.C. Two-scale difference equations. II. Local regularity, infinite products of matrices, and fractals, SIAM J. Math. Anal., 23 (4), pg. 1031-1079, 1992.

[17] Durrande, N., Ginsbourger, D., Roustant O., Carraro, L., ANOVA kernels and RKHS of zero mean functions for model-based sensitivity analysis, Journal of Multivariate Analysis, Vol 155, pg. 57-67, 2013,

[18] Duvenaud, D., Nickisch, H., Rasmussen, C. Additive Gaussian Processes Neural Information Processing Systems, 2011.

[19] Duvenaud, D. PhD Thesis: Automatic Model Construction with Gaussian Processes. http://mlg.eng.cam.ac.uk/duvenaud/; 2014.

[20] http://www.engineeringtoolbox.com/

[21] Fabre J. and Masbernat L. and Suzanne C., Experimental data set no. 7: Stratified flow, part I: Local Structure, Multipahse Science and Technology, Vol. 3, Issue 1-4, pg. 285–301, 1987.

[22] Fan, J. and Gijbels, I. Local Polynomial Modelling and Its Applications. Chapman and Hall, 1996.

[23] Francisco-Fernández, M. and Opsomer, J. D. Smoothing parameter selection methods for nonparametric regression with spatially correlated errors. Canadian Journal of Statistics, 33, pg. 279-295, 2005.

[24] Franco, J. Planification d'expériences numériques en phase exploratoire pour la simulation des phénomènes complexes, PHD thesis, EMSE, 2008.

[25] http://www.kernel-machines.org/

[26] Ginsbourger D., Nicolas D. and Olivier R. Kernels and designs for modelling invariant functions: From group invariance to additivity. mODa 10 - Advances in Model-Oriented Design and Analysis. Contributions to Statistics , pg. 107-115, 2013.

[27] Hastie, T., Tibshirani, R. and Friedman, J. The Elements of Statistical Learning Data Mining, Inference, and Prediction, Second Edition Series: Springer Series in Statistics 2nd ed., 2009.

[28] Janon, A. Analyse de sensibilité et réduction de dimension. Application à l'océanographie. Analysis of PDEs. Université de Grenoble, 2012.

[29] Kaufman, C.G. and Shaby, B.A. The role of the range parameter for estimation and prediction in geostatistics. Biometrika 100(2), pg. 473–484, 2013.

[30] Kennedy, M and O'Hagan, A. Predicting the output from a complex computer code when fast approximations are available. Biometrika. 87(1): pg.1-13; 2000.

[31] Koller, D. and Friedman N. Probabilistic Graphical Models: Principles and Techniques (Adaptive Computation and Machine Learning series) Hardcover – July 31, 2009.

[32] Lam, T-K. and Loh, W-L. Estimating structured correlation matrices in smooth Gaussian random field models Ann. Statist. Volume 28, Number 3, pg. 657-960, 2000.

[33] Le Gratiet, L. Recursive co-kriging model for Design of Computer experiments with multiple levels of fidelity with an application to hydrodynamic, International Journal for Uncertainty Quantification, 4 (5), pg. 365–386, 2014.

[34] Le Gratiet, L. Bayesian analysis of hierarchical multifidelity codes. SIAM/ASA J. Uncertainty Quantification, 1(1), pg. 244– 69; 2013.

[35] Le Gratiet, L. Multi-fidelity Gaussian process regression for computer experiments. Université Paris-Diderot - Paris VII, 2013.

[36] Hecht, F. New development in FreeFem++ J. Numer. Math., Journal of Numerical Mathematics, vol. 20, n. 3-4, pg. 251-265, 2012.

[37] Liu, X.-H. Kernel smoothing for spatially correlated data, 2001.

[38] Mallat, S. G. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. IEEE Trans. Pattern Anal. Mach. Intell. 11, 7, pg. 674-693, 1989.

[39] Micchelli, C. Xu Y., and Zhang, H. Universal kernels. Journal of Machine Learning Research, 7, pg. 2651–2667, 2006.

[40] Opsomer, J., Wang, Y. and Yang, Y. Nonparametric regression with correlated errors. Statistical Science, pg. 134-153. 2001.

[41] Osgood B. G., EE261 The Fourier Transform and its Applications, Fall 2007. http://see.stanford.edu/see/courseInfo.aspx?coll=84d174c2-d74f-493d-92ae-c3f45c0ee091

[42] Qian Z., Seepersad C. C. , Joseph V. R., Allen J. K. and Wu. J. Building surrogate models based on detailed and approximate simulations. AMSE, Vol.128, 2006.

[43] Qian, Z and Wu, J. Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments. Technometrics, 2, Vol.50, 2008.

[44] Quiñonero-Candela, J. and Rasmussen, C.E. A unifying view of sparse approximate Gaussian process regression. Journal of Machine Learning Research, 6, pg. 1939–1959, 2005.

[45] Radford, N. Bayesian learning for neural networks. PhD thesis, University of Toronto, 1995.

[46] Ramsay, J. and Silverman, B. Applied Functional Data Analysis: Methods and Case Studies New York: Springer-Verlag, 2002.

[47] Rasmussen, C E and Williams, C K I. Gaussian processes for machine learning. The MIT press, 2006.

[48] Selvadurai, A.P.S. Partial Differential Equations in Mechanics 1. Springer, 2000.

[49] Snoek J., Larochelle H., Adams R. P. Practical Bayesian Optimization of Machine Learning Algorithms. Advances in Neural Information Processing Systems 25, 2012.

[50] Eric J. Stollnitz, Tony D. Derose, and David H. Salesin. Wavelets for Computer Graphics: Theory and Applications. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[51] Strang, G. 18.06 Linear Algebra, Spring 2010. (MIT OpenCourseWare: Massachusetts Institute of Technology), http://ocw.mit.edu/courses/mathematics/18-06-linear-algebra-spring-2010 (Accessed 22 Jul, 2014). License: Creative Commons BY-NC-SA

[52] Strang, G. 18.085 Computational Science and Engineering I, Fall 2008. (MIT OpenCourseWare: Massachusetts Institute of Technology), http://ocw.mit.edu/courses/mathematics/18-085-computational-science-and-engineering-i-fall-2008 (Accessed 22 Jul, 2014). License: Creative Commons BY-NC-SA

[53] Van Der Vaart, A., Van Zanten, H. Information rates of nonparametric Gaussian process methods. The Journal of Machine Learning Research, 12, pg. 2095-2119, 2011.

[54] Wasserman L. All of statistics: a concise course in statistical inference. Springer, New York, 2004.

[55] Ying, Z. Maximum Likelihood Estimation of Parameters under a Spatial Sampling Scheme Ann. Statist. Volume 21, Number 3, pg. 1119-1662, 1993.

[56] Zhang, H. Inconsistent Estimation and Asymptotically Equal Interpolations in Model-Based Geostatistics Journal of the American Statistical Association, Vol. 99, No. 465, pg. 250-261, 2004.