ENSAI École nationale de la statistique et de l'analyse de l'information

SAFRAN

Louis Allain
3rd year student

Master Thesis

Kernel Methods for Uncertainty Quantification in Machine Learning Predictions

Safran Tech
Rue des jeunes bois, Chateaufort, 78114 Magny les Hameaux, France

Graduate Year: 2024

Supervisor: Brian Staber

Teacher: Sébastien Da Veiga

Abstract

In this work, we focus on the construction of prediction intervals for machine learning models. Numerical simulation is widely used to predict the behavior of industrial pieces. In most industrial applications, each call to the simulation code can take hours, making direct optimisation intractable. To overcome this issue, machine learning has been exploited to build surrogate models for the simulation code. While these machine learning models proved effective across several industries, there is a lack of rigorous uncertainty quantification for the prediction made by those models.

Lately, the theory of conformal prediction has attracted a lot of attention. Indeed, it provides a framework to build coverage guarantees in finite samples, without making any assumptions on the true distribution of the data. Yet, conformal prediction suffers from, first, its central assumption, exchangeability, which makes it incompatible with design of experiments. Second, despite active research, the prediction bands are not really adaptative.

T. Liang (2022) and Fan et al. (2024) propose a novel framework to build prediction bands, capitalising on kernel methods. We present a thorough investigation of their propositions and grasp the limitations of their framework. Thus, we suggest improvements to make the model tractable and consistent across our toy experiments. Finally, we explain why this framework can be easier to generalise to Safran's usecases and we develop on research directions.

Contents

List of Figures **7**

List of Tables **9**

List of Algorithms **11**

List of Symbols **11**

Introduction **15**

1 Research at Safran **17**

2 Machine learning models to quantify uncertainty **19**
2.1 All about coverage 19
2.2 Prediction intervals in statistics 20
2.3 Ensemble learning 20
2.3.1 A wide variety of models 21
2.3.2 Building prediction bands 22
2.4 Quantile regression 22
2.5 Gaussian processes 23
2.5.1 Definiton 23
2.5.2 Training 24
2.5.3 Inference 24

3 Conformal prediction **27**
3.1 A naive idea 27
3.2 Split conformal prediction 28
3.3 Jackknife+ 30
3.4 Discussion about adaptativity 31
3.4.1 Rescaling 31
3.4.2 Conformalized quantile regression 32

4 Kernel methods to assess uncertainty in the predictions **35**
4.1 Minimising the nuclear norm 35
4.1.1 Introduction to the semi-definite problem 35
4.1.2 Empirical investigation 38
4.2 Minimising the average width 40
4.2.1 General setting 40
4.2.2 A special case with a RKHS 42
4.2.3 Illustration 43

5 Improving kernel methods **47**
5.1 Representation of non-negative functions 47
5.1.1 A representer theorem 47
5.1.2 Equivalent formulation of the primal 48
5.1.3 Using this equivalent form 49

5.1.4 On the regulariser function 49
5.2 Refined calibration 51
5.3 Illustrative comparison 51
5.4 Hyperparameter tuning 55

6 Kernel methods to estimate the mean and variance at once 61
6.1 Conjectures for a representer theorem 61
6.2 Empirical study 62

Conclusion 65

Prospects 67

A Experiments description 73

B All figures details 75

C Experiments complementary results 77

D Conformal predictions 81

E More on the simultaneous model 93

List of Figures

2.1 Experiment 4 - Random Forest. 21
2.2 Experiment 4 - Kernel Quantile Regression. 22
2.3 Experiment 4 - Gaussian Process. 24

3.1 Experiment 4 - Split Conformal Prediction. 29
3.2 Experiment 4 - Jackknife+. 31
3.3 Experiment 4 - J+GP model. 32
3.4 Experiment 4 - Kernel Conformalized Quantile Regression. 33

4.1 Experiment 2 - Liang B - $\Delta = 1$ 39
4.2 Experiment 2 - Liang B - $\Delta = 10$ 39
4.3 Experiment 4 - Liang B - $\Delta = 1$ 40
4.4 Experiment 2 - Fan B - $\delta = 0.001$; $p = 2$ 43
4.5 Experiment 2 - Fan B - $\delta = 0.001$; $p = 100$ 44
4.6 Experiment 4 - Fan B - $\delta = 0.001$; $p = 100$ 44
4.7 Experiment 4 - Fan B - $ls = 0.6$; $p = 100$ 45

5.1 Time comparison between formulation A, B and Frobenius regularisation. 49
5.2 Time comparison between different regularisation values λ_2. 50
5.3 Experiment 2 - Liang A - $\lambda_2 = 1$; $\delta = 0.001$. 52
5.4 Experiment 2 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2, 100$ 53
5.5 Experiment 4 - Liang A - $\lambda_2 = 1$; $\delta = 0.001$ 53
5.6 Experiment 4 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2, 100$. 54
5.7 Experiment 4 - Liang A - $ls = 0.2$; $\delta = 0.001$. 55
5.8 Experiment 4 - Liang A - $ls = 0.2$; $\lambda_2 = 1$. 56
5.9 Pearson coefficient between residuals and prediction bands width, on calibration data. 56
5.10 Spearman coefficient between residuals and prediction bands width, on calibration data. 57
5.11 Mutual Information between residuals and prediction bands width, on calibration data. 57
5.12 Energy-HSIC between residuals and prediction bands width, on calibration data. 58

6.4 Energy-HSIC between residuals and prediction bandss width, on calibration data in Figure 6.3. 63
6.1 Experiment 2 - Liang A, simultaneous - $\lambda_2 = 1$;$\delta = 0.001$. . 63
6.2 Experiment 2 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$;$p = 2$. 63
6.3 Experiment 4 - Liang A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$. . 64
6.6 Energy-HSIC between residuals and prediction bandss width, on calibration data, on a finer scale. 64
6.5 Experiment 4 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2$. 64

6.7 Experiment 4 - Simultaneous - $\lambda_2 = 1$; $\delta = 0.001$. 64
A.1 First toy dataset. 73
A.2 Second toy dataset. 73
A.3 Third toy dataset. 73

A.4 Fourth toy dataset. 74

C.1 Experiment 1 - Liang A - $\lambda_2 = 1$;$\delta = 0.001$. 77
C.2 Experiment 1 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$. 78
C.3 Experiment 1 - Liang A, simultaneous - $\lambda_2 = 1$;$\delta = 0.001$. . 78
C.4 Experiment 1 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$. 78
C.5 Experiment 3 - Liang A - $\lambda_2 = 1$;$\delta = 0.001$. 79
C.6 Experiment 3 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$. 79
C.7 Experiment 3 - Liang A, simultaneous - $\lambda_2 = 1$;$\delta = 0.001$. . 80
C.8 Experiment 3 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$. 80

E.1 Experiment 4 - Liang, simultaneous estimation with quadratic constraints - $\lambda_2 = 1$;$\delta = 0.001$. 94
E.2 Experiment 4 - Fan, simultaneous estimation with quadratic constraints - $\lambda_2 = 1$;$\delta = 0.001$. 94
E.3 Energy-HSIC between residuals and prediction bands's width (on calibration data) in figure E.1 and E.2, meaning, with quadratic constraints. 94
E.4 Experiment 4 - Liang, simultaneous - $ls = 0.2$;$\delta = 0.001$. . . 95
E.5 Experiment 4 - Liang, simultaneous - $ls = 0.4$;$\delta = 0.001$. . . 95
E.6 Experiment 4 - Fan, simultaneous - $ls = 0.2$;$\delta = 0.001$. . . . 95

List of Tables

4.1 Statistics about experiments in Figure 4.1. 38
4.2 Statistics about experiments in Figure 4.2. 39
4.3 Statistics about experiments in Figure 4.4. 43
4.4 Statistics about experiments in Figure 4.5. 44

5.1 Statistics about experiments in Figure 5.3. 52
5.2 Statistics about experiments in Figure 5.4. 52
5.3 Statistics about experiments in Figure 5.5. 53

List of Algorithms

1 Calibration for Coverage Control in T. Liang (2022) 38
2 Calibration for Coverage Control in Fan et al. (2024) 41

3 Calibration for Coverage Control 51

List of Symbols

$\mathcal{X}$	Input space.
$\mathcal{Y} \subset \mathbb{R}$	Output space.
d	Number of features.
$\boldsymbol{K}_x$	Column vector of the kernel matrix $\boldsymbol{K}$ associated to observation x.
$\boldsymbol{X}$	Feature matrix.
$\boldsymbol{y}$	Output vector.
$\langle \cdot, \cdot \rangle$	Scalar product in $\mathbb{R}^n$.
$\mathbb{S}^n$	Set of symmetric matrices of size $n \times n$.
$\mathcal{H}_k$	Reproducing Kernel Hilbert Space associated to kernel k.
$\mathcal{S}(\mathcal{H})$	Set of bounded Hermitian operators on space $\mathcal{H}$.
$\overline{\mathbb{R}}$	$\mathbb{R} \cup \{\pm\infty\}$
ϕ_k	Feature map associated to kernel k.
$k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$	Kernel function.
$k(\boldsymbol{X}_1, \boldsymbol{X}_2) = \boldsymbol{K}_{\boldsymbol{X}_1 \boldsymbol{X}_2}$	Kernel matrix or Gram matrix.
x, y	One observation of features and output.
$\hat{m} \colon \mathcal{X} \to \mathcal{Y}$	Estimation of the conditional mean function.
$\hat{v} \colon \mathcal{X} \to \mathcal{Y}$	Estimation of the conditional variance function.
$\widehat{C} \colon \mathcal{X} \to \text{subset}(\mathcal{Y})$	Estimation of the prediction bands.

Introduction

To pursue its effort for better and cleaner aviation, Safran is on the forefront of research. The company is especially trying to leverage the power of machine learning advances into the development and conception of new motor pieces.

Today, Safran heavely relies on numerical simulations to predict the behaviour of industrial pieces in order to identify designs that meet the desired performances. Finding the appropriate design (in terms of geometry and material, for instance) typically involves solving an optimisation problem, requiring sometimes several hundreds of calls to the simulation code that predicts the underlying physics. In most industrial applications, each call to the simulation code can take several hours to properly model the physics. For this reason, direct optimisation is most of the time intractable. To overcome this issue, a common practice consists in building a surrogate model for the simulation code. Popular surrogate models include Gaussian processes, random forests, or, more recently, neural networks for their high flexibility. While these machine learning models proved effective across several industries, there is a lack of rigorous uncertainty quantification for the predictions made by such models.

Recently, there has been a growing interest in conformal prediction methods which provide a systematic way to build prediction intervals with coverage guarantees, see Vovk et al. (2005) and Barber et al. (2020a). Conformal prediction is a distribution-free and model agnostic framework that provides prediction intervals with marginal coverage guarantees and also, for some variants, training-conditional coverage guarantees at the expense of additional assumptions. Nevertheless, it still suffers from a few shortcomings such as the incompatibility with deterministic design of experiments due to the exchangeability assumption, or its tendency to generate rather constant intervals. Although many effort were made into developing adaptive conformal prediction bands (Lei et al. 2017; Romano et al. 2019), it still remains an active research topic.

Here, we propose to investigate the kernel framework recently proposed by T. Liang (2022) and Fan et al. (2024) that rely on semi-definite programming. The methodology consists in learning the conditional variance of the output given the input features. This function is learned in a complex and high dimensional space using a positive definite kernel and the representer theorem for non-negative functions (Marteau-Ferey et al. 2020). The generated prediction intervals also enjoy marginal and training-conditional coverage guarantees without relying on the exchangeability assumption, which appears promising for handling deterministic design of experiments often used in numerical simulations.

Throughout this work, the performance of kernel methods (T. Liang 2022; Fan et al. 2024) have been found promising and it outperforms conformal prediction variants on several synthetic regression problems, especially in terms of adaptivity. We also show that the optimisation

problems defined in T. Liang (2022) and Fan et al. (2024) can be casted into a single framework, and that the computational times can be greatly reduced by relying on results presented in Marteau-Ferey et al. (2020). Finally, we also propose a completely novel methodology for tuning the lengthscales involved in the kernel methods, which was highly overlooked in T. Liang (2022) and Fan et al. (2024).

In Chapter 1, we briefly present the company and the master thesis environment.

In Chapters 2 and 3, we present traditional machine learning methods and their way of constructing prediction bands. We understand why those methods cannot be trusted and used for industrial applications. Next, reviewing recent advances on conformal prediction theory, we shall see that this theory is not satisfactory enough for real applications.

We will then introduce the kernel method in Chapter 4. The propositions of T. Liang (2022) and Fan et al. (2024) are promising but lack a few key points. First the calibration procedures are inefficient and second, the methods are left with numerous hyperparameters to tune.

In Chapters 5 and 6, we provide our propositions for improvement. First, we leverage the work of Marteau-Ferey et al. (2020) to improve the training times and understand that both kernel models can be casted into a single one. Second, we propose a calibration method that is easy to use and provide both theoretical guarantees and empirical performances. Third, we overcome an oversight of both Liang and Fan, the tuning of the kernel length scale, with satisfactory results. Finally, we leverage all this work into a model that estimates both the mean and variance functions.

Finally, we discuss further research directions to improve the model and expand it to Safran usecases.

Chapter 1

Research at Safran

Safran is the result of the merge of two important french companies Snecma (*Société nationale d'étude et de construction de moteurs d'aviation*) and Sagem (*Société d'applications générales d'électricité et de mécanique*). The merge happened in 2005 and since then, Safran has imposed itself as a leader in fields such as aeronautics, aerospace and defence. It is the third biggest aeronautic company, employs 92,000 people around the world and generated more than 23 billions euros in 2023. The five biggest activities of Safran are aircraft engines, aeronautics equipments, interiors designs, defence and spatial technologies.

Fortified by all these activities, Safran must stay at the leading edge of technology. For this purpose, Safran Group created in 2015 the Safran Tech structure, which is the group's research and technology center. Safran Tech brings together 550 researchers. It organises itself around three main research areas. **Materials and processes**, which already provided groundbreaking technologies such as carbon brakes and composite rotor blades. **Energy and propulsion**, which gathers the research on energy propulsion and electronics systems. And, lastly, **information**, which focuses on sensors and measure systems, computer experiments and simulations, as well as exploring machine learning tools to assist conception and simulation.

Our internship takes place in that very last area. Specifically, during conception of industrial pieces, numerical simulation is widely used. Those simulations are heavy and can take several hours to compute. A machine learning solution consists in training a surrogate model on a few hundred solutions, in order to predict for a new industrial piece the outcome of the numerical simulation quicker. This solution proved effective across several industries. However, people are reluctant to those models because of the lack of rigorous uncertainty quantification in their predictions. This internship is a step towards better uncertainty quantification for machine learning predictions.

Chapter 2

Machine learning models to quantify uncertainty

Traditional machine learning models are estimating a *point predictor* regression function. There are no, or very little, information about the reliability of this point prediction. In this chapter we are going to review the notion of coverage for prediction intervals and then popular methods in machine learning that quantifies uncertainty in the predictions. Most methods, to quantify uncertainty, rely on heavy assumptions about the data. Those assumptions are often unfounded and thus make the mathematical background on which they rely unsuitable.

2.1 All about coverage

In order to quantify the reliability of the traditional machine learning model, one way is to construct a machine learning model that trains a set predictor rather than a point predictor. This set predictor produces prediction intervals or prediction bands, often around a point predictor. Given a dataset $\mathcal{D}_n$, we write the prediction bands trained on $\mathcal{D}_n$, $\widehat{C}_n \colon \mathcal{X} \to \text{subset}(\mathcal{Y})$. In order for those prediction bands to make sense, we must have confidence in them too. The way to have prediction bands that are reliable, is to construct those bands in a way that mathematically guarantees their correctness. This correctness property is transcribed through what we call the *coverage*. Therearfter we define three different type of coverage that we could be interested in. We denote by $\alpha \in (0,1)$ the error rate.

Definition 1 (Marginal coverage)**.** *Let $\alpha \in (0,1)$ and $\mathcal{D}_n$ be the dataset used for training the set predictor $\widehat{C}_n$. We say that $\widehat{C}_n$ satisfies a marginal coverage guarantee if we have:*

$$\mathbb{P}\left(Y \in \widehat{C}_n(X)\right) \geq 1 - \alpha \tag{2.1}$$

where the probability $\mathbb{P}$ is computed over the data: $\mathcal{D}_n \cup (X, Y)$.

Definition 2 (Train-conditional coverage)**.** *Let $\alpha \in (0,1)$ and $\mathcal{D}_n$ be the dataset used for training the set predictor $\widehat{C}_n$. We say that $\widehat{C}_n$ satisfies a train-conditional coverage guarantee if we have:*

$$\mathbb{P}\left(Y \in \widehat{C}_n(X) \,|\mathcal{D}_n\right) \geq 1 - \alpha \tag{2.2}$$

where the probability $\mathbb{P}$ is computed over the data: $\mathcal{D}_n \cup (X, Y)$.

Training-conditional coverage is interesting, because it states that for every random draw of training sample, the coverage guarantee is met. This

is much stronger than the marginal coverage which states that, on average on the training samples the coverage is met. This means that for a given training sample, the prediction bands might not satisfy the coverage. But if we trained prediction bands for many different training samples, on average the coverage would be met. The relationship between the marginal coverage and the train-conditional one is as follows:

$$\mathbb{E}_{\mathcal{D}_n}\left[\mathbb{P}\left(Y \in \widehat{C}_n(X) \,|\mathcal{D}_n\right) \geq 1-\alpha\right] = \mathbb{P}\left(Y \in \widehat{C}_n(X)\right) \geq 1-\alpha \quad (2.3)$$

Definition 3 (Test-conditional coverage)**.** *Let $\alpha \in (0,1)$ and $\mathcal{D}_n$ be the dataset used for training the set predictor $\widehat{C}_n$. We say that $\widehat{C}_n$ satisfies a test-conditional coverage guarantee if we have:*

$$\mathbb{P}\left(Y \in \widehat{C}_n(X) \,|X = x\right) \geq 1-\alpha \quad (2.4)$$

where the probability $\mathbb{P}$ is computed over the data: $\mathcal{D}_n \cup (X,Y)$.

This coverage is stronger than the previous two. It is linked to the adaptativity of the prediction bands. Indeed, completely adaptative bands would satisfy test-conditional coverage and vice-versa. Unfortunately, this test-conditional coverage comes with a theorem which states that if no assumption is placed on the distribution of the data, then any prediction bands $\widehat{C}_n(\cdot)$ satisfying Definition 3 would have infinite length, which is of course not desirable, see Barber et al. (2020b).

2.2 Prediction intervals in statistics

In statistics prediction intervals can be built when considering the linear Gaussian models, which writes:

$$\boldsymbol{y} = \boldsymbol{X}\beta + \varepsilon \quad (2.5)$$

where the error term ε is centered, homoscedastic, uncorrelated and follows a Gaussian distribution: $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 \mathrm{Id}\right)$.
From this model, one can construct confidence intervals. Those are not what we are interested in because *confidence intervals* are about the coefficients β of the model.
Still, we can construct *prediction intervals* or *prediction bands* for each and every prediction. Suppose we trained the model and computed $\widehat{\beta}$ using a training data $\mathcal{D}_n$. We now want to predict at a new datum (x,y). First we need to suppose that $y = x^\top \beta + \eta$ where $\eta \sim \mathcal{N}\left(0, \sigma^2\right)$ and is independent from ε. Only then can we construct the prediction interval around our prediction $x^\top \widehat{\beta}$:

$$\widehat{C}_n(x) = \left[x^\top \widehat{\beta} \pm t_{n-d-1}(1-\alpha/2)\widehat{\sigma}\sqrt{x^\top \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} x + 1}\right] \quad (2.6)$$

where $\widehat{\sigma}$ is an estimation of the error's variance and $t_{n-d-1}(1-\alpha/2)$ is the $1-\alpha/2$ quantile of the Student's distribution of $n-d-1$ degrees of freedom.

It is clear that the Gaussian assumption on the error and the parametric nature of the model makes it unsuitable for efficient uncertainty quantification in more complex setup.

2.3 Ensemble learning

We call here an ensemble method any machine learning model for which the final predictor is an aggregation of numerous predictors. The idea

behind ensemble learning is that combining several models will achieve better accuracy than a single one. It follows the idea of the "wisdom of the crowd". Now we will briefly go over a few well known models and then develop how, from an ensemble method one can construct prediction bands.

2.3.1 A wide variety of models

We present here as an ensemble method any method that for final prediction computes the mean of other models. Some models presented here are not part of what falls traditionally under the "ensemble" name. However we can compute prediction bands in a similar fashion, and that is why we present them together, for the sake of brevity.

Bagging. In its simplest form **B**ootstrap **Agg**ragat**ing** trains the same model on different samples of the training data. The samples are created by Bootstrap (see Breiman (1996)), and aims at mimicking multiple independent training data. This can be applied to any machine learning model[1], but the most famous bagging technique is Random Forest. It has been proposed by Breiman (2001) and the idea is to simultaneously train B trees on different bootstrap samples. To further increase the diversity of the trees within the forest, multiple techniques are used. First, splits are decided on a random subset of the features. Second, the trees are not pruned to favor overfitting. For illustration purposes, Figure 2.1 presents a random forest model on experiment four, more details in Appendix B. By training B CART trees this way, the random forest prediction at $x \in \mathcal{X}$ is

$$\widehat{m}_B(x) = \frac{1}{B}\sum_{b=1}^{B} m_b(x). \tag{2.7}$$

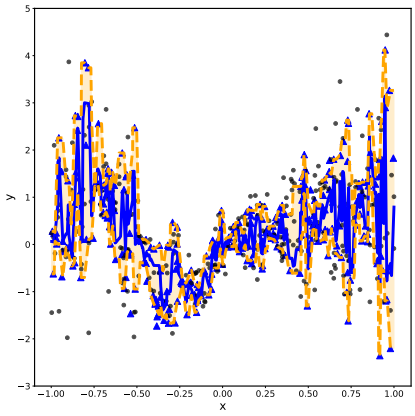


Figure 2.1: Experiment 4 - Random Forest model with prediction bands computed as in Equation (2.14).

Boosting. Here we do not rely on bootstrap samples and the fundamental idea is to train the models *sequentially* rather than simultaneously in bagging models. Boosting methods rely on the fact that we can evaluate the errors of the model at every aggregation step. You can either use the errors to grade the given learner (grade which will be used as a weight in the aggregation) and to re-weight the training samples that are not predicted accurately, in order to indicate to the following learner that theses observations are more relevant to him. A very popular method is AdaBoost. Or, we can use the errors to train the following learner, by using them as the target variable. A wide variety of methods exists, a few popular ones are Gradient Boosting, XGBoost, lightGBM and CatBoost. In boosting methods, if B is the total number of iterations, the final prediction at $x \in \mathcal{X}$ is computed as

$$\widehat{m}_B(x) = \sum_{b=1}^{B} \gamma_b m_b(x). \tag{2.8}$$

Bayesian Neural Networks. As the two previous methods are generally implemented using CART trees, here we are going to leverage neural networks. Without going in the details of neural networks, Bayesian Neural Networks works by assigning distributions to all parameters of the network. During inference, instead of having a single network, we sample the parameters from the posterior distributions and aggregate the prediction of all those networks. If we denote by m_b the predictor of a

[1] To promote independence and diversity it is usually used with *weak learners* machine learning models, such as CART trees, which predictions are very unstable.

network for one sample of all parameters, and say we sample the weights B times, for a new data point $x \in \mathcal{X}$, the final predictor writes

$$\widehat{m}_B(x) = \frac{1}{B}\sum_{b=1}^{B} m_b(x). \tag{2.9}$$

Deep ensemble. What is called deep ensemble is simply training a variety of neural networks on the same data but with different initialisation, different random seeds or different noise values in the stochastic gradient descent algorithm. Say we perform B different initialisation, for $x \in \mathcal{X}$ we predict

$$\widehat{m}_B(x) = \frac{1}{B}\sum_{b=1}^{B} m_b(x). \tag{2.10}$$

2.3.2 Building prediction bands

All ensemble methods have a final predictor that can be written the following way, considering the training data $\mathcal{D}_n$ and if we denote by B the number of different "sub models", for $x \in \mathcal{X}$:

$$\widehat{m}_B(x) = \sum_{b=1}^{B} \gamma_b m_b(x) \tag{2.11}$$

From here, all ensemble methods can construct prediction bands, at level α, by computing the quantiles $\alpha/2$ and $1-\alpha/2$:

$$\widehat{q}_{B,\alpha/2}(x) = Q_{\alpha/2}\left(\sum_{b=1}^{B} \gamma_b \delta_{m_b(x)}\right) \tag{2.12}$$

$$\widehat{q}_{B,1-\alpha/2}(x) = Q_{1-\alpha/2}\left(\sum_{b=1}^{B} \gamma_b \delta_{m_b(x)}\right) \tag{2.13}$$

Finally, we have:

$$\widehat{C}_n(x) = \left[\widehat{q}_{B,\alpha/2}(x), \widehat{q}_{B,1-\alpha/2}(x)\right] \tag{2.14}$$

Using this method allows to compute prediction bands instantaneously around the prediction of the ensemble method. For convergence of those estimators to the true quantile see Meinshausen et al. (2006) for random forests. Ensemble learning methods can be powerful empirically, but they do not satisfy any coverage property discussed in the previous section.

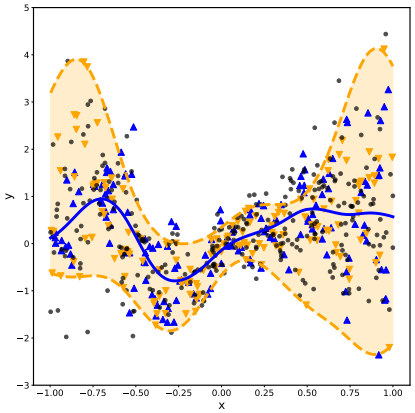


Figure 2.2: Experiment 4 - Kernel Quantile Regression model with a Gaussian process to estimate the mean function and two kernel quantile regression to estimate the prediction bands, using a Matérn kernel.

2.4 Quantile regression

A third way to construct prediction bands is to use the pinball loss as a drop in replacement of the usual quadratic loss. This pinball loss allows to estimate the quantile function at a specified level, rather than the mean function. It writes, for a level $\alpha \in (0,1)$:

$$L_\alpha(y, y') = (y' - y)\left(\mathbb{1}_{(y \le y')} - \alpha\right). \tag{2.15}$$

It is indeed a loss function as it verifies the following two properties:

$$\text{i) } \forall\, y \in \mathcal{Y},\ L_\alpha(y,y) = 0 \quad \text{ii) } \forall\, (y,y') \in \mathcal{Y}^2,\ L_\alpha(y,y') > 0 \tag{2.16}$$

Because of this drop in replacement, any usual machine learning models in which the quadratic loss is minimised can be used to estimate the quantiles:

[2] Boosting is also considered an ensemble method. But the way those models are trained allows the use of the pinball loss. To create prediction bands boosting can be used either as en ensemble model or as a quantile regression model.

linear regression, kernel ridge regression, neural networks, splines, boosting models[2]. To compute the prediction bands at level $\alpha \in (0,1)$, the machine learning model should be trained twice, one at level $\alpha/2$ to estimate $\widehat{q}_{\alpha/2}$ and once at level $1-\alpha/2$ to estimate $\widehat{q}_{1-\alpha/2}$.

The prediction bands are then computed as, for $x \in \mathcal{X}$ and $\mathcal{D}_n$ a training dataset:

$$\widehat{C}_n(x) = \left[\widehat{q}_{\alpha/2}(x), \widehat{q}_{1-\alpha/2}(x)\right] \tag{2.17}$$

As an example we showcase a kernel quantile regression model in Figure 2.2, more details in Appendix B. Once again there exists results that shows the convergence of those estimators to the true quantiles. It can be shown that quantile regression can achieve coverage, but only as the number of training sample goes to infinity, therefore not in finite sample. Some recent work are investigating new methods, that do not involve the pinball loss, to estimate conditional quantiles, see Chung et al. (2021).

2.5 Gaussian processes

Gaussian processes were first introduced in geographic and geostatistics fields under the name of *kriging*, Krige (1951) and Matheron (1962). In the machine learning community this model is known as Gaussian processes (Rasmussen et al. 2006). Because we are going to make use of Gaussian processes in following chapters, we present it in more details than the previous three sections.

2.5.1 Definiton

Before investigating the proper Gaussian process distribution let us review the classic Gaussian (normal) distribution and some of its properties. A Gaussian distribution of dimension $d \in [\![1, n]\!]$ has a probability density function given by:

$$f(x) = \frac{1}{2\pi^{d/2}|\Sigma|} e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)} \tag{2.18}$$

where $\mu \in \mathbb{R}^d$ is the mean vector and $\Sigma \in \mathbb{R}^{d\times d}$ is the covariance matrix. For X a random variable, we write X follows a Gaussian distribution

$$X \sim \mathcal{N}(\mu, \Sigma).$$

A few properties that caracterises the Gaussian distribution are:

- Gaussian distributions are stable for the sum, the product and affine transformations.
- All marginal distributions are still Gaussian distributions.
- Conditioning by a Gaussian distribution is still a Gaussian distribution.

Gaussian processes extend this Gaussian distribution to an infinite collection of variables. Consider $X = (X_1, X_2, \ldots)$ a random process that follows a Gaussian process distribution:

$$X \sim \mathcal{GP}(\mu(\cdot), k(\cdot,\cdot)) \tag{2.19}$$

with mean function $\mu(\cdot)$ and covariance function $k(\cdot,\cdot)$. The main property of Gaussian processes is that any finite subset of this process follows a Gaussian distribution. Therefore for a dataset $\mathcal{D}_n = (\boldsymbol{X}, \boldsymbol{y}) \in (\mathcal{X}\times\mathcal{Y})^n$, assuming that this data comes from a Gaussian process distribution, $\boldsymbol{y}$ follows a Gaussian distribution of mean vector $\mu(\boldsymbol{X})$ and of covariance matrix $k(\boldsymbol{X}, \boldsymbol{X})$. A formal definition given in Rasmussen (2004) is:

Definition 4 (Gaussian process)**.** *A Gaussian Process is a collection of random variables, any finite number of which have (consistent) joint Gaussian distribution.*

2.5.2 Training

The Gaussian process model is used within a Bayesian framework, the Gaussian process distribution being used as a prior for Bayesian inference. The specification of the mean function and the covariance function represents some prior information. Denote θ the set of parameters involved in those functions, they are hyperparameters of the model. The training of the Gaussian process model refers to finding the best set of hyperparameters in the light of the data. To do that, we maximise the log marginal likelihood of the model, with regard to the data:

$$\log p\left(\boldsymbol{y}|\boldsymbol{X},\theta\right) = -\frac{1}{2}\log(|\Sigma|) - \frac{1}{2}\left(\boldsymbol{y}-\mu\right)^{\top}\Sigma^{-1}\left(\boldsymbol{y}-\mu\right) - \frac{n}{2}\log(2\pi).$$

In practice the maximisation is usually done using optimisation algorithm such a L-BFGS. Once the optimal hyperparameters $\widehat{\theta}$ are found, we can use this Gaussian process model to infer on new data points.

2.5.3 Inference

Once the best hyperparameters for the training data are obtained, it is straightforward to predict for unseen test cases. Let $x \in \mathcal{X}$ be a new data point and y its associated true output value. From Definition 4, the joint distribution of the training data and the unseen data follows a Gaussian distribution.

$$\begin{bmatrix}\boldsymbol{y}\\ y\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\mu(\boldsymbol{X})\\ \mu(x)\end{bmatrix}, \begin{bmatrix}k(\boldsymbol{X},\boldsymbol{X}) & k(\boldsymbol{X},x)\\ k(\boldsymbol{X},x) & k(x,x)\end{bmatrix}\right) \tag{2.20}$$

Because we know the values of the training cases, we are interested in the conditional distribution of y knowing $\boldsymbol{y}$:

$$y|\boldsymbol{y} \sim \mathcal{N}\left(\widehat{m}_n(x), \widehat{v}_n(x)\right). \tag{2.21}$$

The prediction of the Gaussian process is the mean of this distribution:

$$\widehat{m}_n(x) = \mu(x) + k(\boldsymbol{X},x)k(\boldsymbol{X},\boldsymbol{X})^{-1}(\boldsymbol{X}-\mu(\boldsymbol{X})). \tag{2.22}$$

Where Gaussian process model goes beyond is that we do not predict a single point but rather a probability distribution around this point. We have access to the predicted variance at each test point:

$$\widehat{v}_n(x) = k(x,x) - k(\boldsymbol{X},x)k(\boldsymbol{X},\boldsymbol{X})k(\boldsymbol{X},x). \tag{2.23}$$

Because we assume a Gaussian process distribution, using this variance we can compute prediction bands:

$$\widehat{C}_n(x) = \left[\widehat{m}_n(x) \pm q_{\alpha/2}\sqrt{\widehat{v}_n(x)}\right] \tag{2.24}$$

Where $q_{\alpha/2}$ is the $\alpha/2$ quantile of the normal distribution. For illustration, Figure 2.3 exposes the performances of a Gaussian Process on experiment four. More details can be found in Appendix B. This model is evidently not distribution free. Moreover, the marginal coverage guarantee is met only if the true underlying variance of the data lives in the Reproducing Kernel Hilbert Space defined by the posterior kernel. Thus Gaussian process do

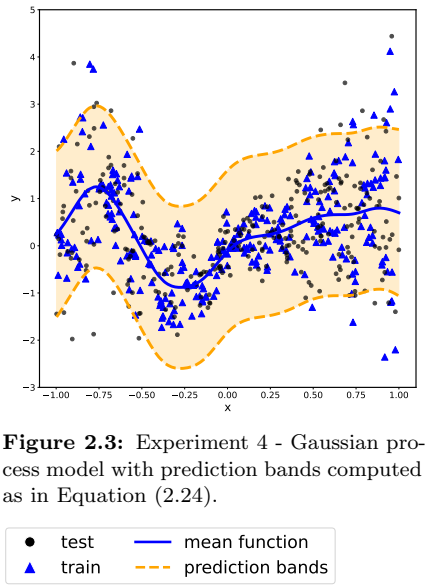


Figure 2.3: Experiment 4 - Gaussian process model with prediction bands computed as in Equation (2.24).

not meet the theoretical guarantees discussed in the previous chapter.

As a matter of fact, to our knowledge, there exists no other framework that allows for distribution free and finite sample coverage guarantee besides conformal prediction.

Chapter 3

Conformal prediction

This theory has emerged as a mathematically strong and practically reliable method for uncertainty quantification in the predictions. It originated from Gammerman et al. (1998) and since then numerous propositions were made to improve the method. Foundation references are Vovk et al. (2005) and Lei et al. (2017) and Vovk (2012a). The theory of conformal prediction relies on the prediction quality of a machine learning model to asses the uncertainty of this prediction, thus yielding a prediction interval rather than a point prediction. In this chapter we will present the main algorithms of conformal prediction and elaborate on adaptativity. In Appendix D we present a more complete overview of conformal prediction.

Conformal prediction aims at quantifying the degree of certainty one has over a prediction of a machine learning model. Formally, for a given error rate $\alpha \in]0,1[$ and a dataset $\mathcal{D}_n$, we want to train a prediction set $\widehat{C}_n \colon \mathcal{X} \to \text{subset}(\mathcal{Y})$ such that for any new observation (x, y) we have:

$$\mathbb{P}\left(y \in \widehat{C}_n(x)\right) \geq 1 - \alpha \tag{3.1}$$

That is the goal of conformal prediction. With that, we also want our model to have a few distinct features that set conformal prediction apart from other models. First, we want Equation (3.1) to hold in *finite sample*. Second, we want the model to be *distribution free*, we do not want any assumptions on the real distribution of the data. Third, we want to have *adaptability* of the prediction bands, meaning we do not want $\widehat{C}_n$ to be of constant width. Fourth, we would prefer a model that is *agnostic* to any machine learning point predictor. Indeed we would like to construct such set around a machine learning model. We now develop on *some* variants of the conformal prediction algorithm, starting with the finite sample correction, which is key in conformal prediction.

3.1 A naive idea

The very first idea one can have to verify Equation (3.1) is to compute the $1 - \alpha$ quantile. Let us see on a simpler case why this idea is too simple. Consider a sequence of observations $S_i \in \mathbb{R}$, $i = 1, \ldots, n$, a test point S_{n+1} and a one sided prediction set $\widehat{C}_n =]-\infty, \widehat{q}_n]$. Our objective Equation (3.1) becomes

$$\mathbb{P}\left(S_{n+1} \in \widehat{C}_n(X_{n+1})\right) = \mathbb{P}\left(S_{n+1} \leq \widehat{q}_n\right) \geq 1 - \alpha \tag{3.2}$$

Recall the definition of a quantile and by choosing $\widehat{q}_n$ as the $1-\alpha$ empirical quantile of the $S_i, i = 1, \ldots, n$,

$$\widehat{q}_n = Q_{1-\alpha}\left(\frac{1}{n}\sum_{i=1}^{n}\delta_{S_i}\right) \tag{3.3}$$

we obtain the approximate result $\mathbb{P}(S_{n+1} \leq \widehat{q}_n) \approx 1-\alpha$, which becomes exact as n tends to infinity

$$\text{for } n \to \infty,\ \mathbb{P}(Y_{n+1} \leq \widehat{q}_n) \geq 1-\alpha \tag{3.4}$$

Our objective here is to obtain the same guarantee in finite sample, we want it to be true for all n and not only when $n \to \infty$.

To achieve this we first need to assume that our observations $S_1, \ldots, S_{n+1}$ are exchangeable[3].

[3] Exchangeability is weaker than independent and identically distributed (iid). If a sample is iid then it is exchangeable.

Definition 5 (Exchangeability)**.** *$S_1, \ldots, S_{n+1}$ are said to be exchangeable if their joint distribution is invariant under permutation:*

$$\forall \sigma \in \mathfrak{S}, (S_1, \ldots, S_{n+1}) \overset{\mathrm{d}}{=} \left(S_{\sigma(1)}, \ldots, S_{\sigma(n+1)}\right) \tag{3.5}$$

where $\mathfrak{S}$ is the set of all possible permutations.

Second we need to adjust the confidence level α to $\frac{\lceil (1-\alpha)(n+1) \rceil}{n}$. Using exchangeability and computing $\widehat{q}_n$ as

$$\widehat{q}_n = Q_{\frac{\lceil (1-\alpha)(n+1) \rceil}{n}}\left(\frac{1}{n}\sum_{i=1}^{n}\delta_{S_i}\right) \tag{3.6}$$

we can achieve $\mathbb{P}(S_{n+1} \leq \widehat{q}_n) \geq 1-\alpha$ in finite sample. The newly defined $\widehat{q}_n$ can be seen as the empirical quantile at an adjusted level to allow for a finite sample guarantee, this is the *finite sample correction*. This adjustment will be key in the theoretical guarantees of conformal prediction methods.

3.2 Split conformal prediction

The very first conformal prediction algorithm was presented by Gammerman et al. (1998) and is referred to as full conformal prediction, transductive conformal prediction or simply conformal prediction. The problem is the computational burden of this method, which makes it unusable in practice. Very early on, Harris Papadopoulos et al. (2002), proposed to split the training dataset to achieve exchangeability of the residuals without the computational burden[4]. Let $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(x_i, y_i) \sim \mathcal{P}$ where $\mathcal{P}$ is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(x_{n+1}, y_{n+1}) \sim \mathcal{P}$ be a test point. First of all, let us *split* the training dataset in two, a proper training dataset $\mathcal{D}_1$ of size n_1 and a calibration one $\mathcal{D}_2$ of size n_2. We begin by training a point predictor regression function[5] on the proper training dataset $\mathcal{D}_1$:

$$\widehat{m}_{n_1} \colon \mathcal{X} \to \mathcal{Y} \tag{3.7}$$

Then, we compute the residuals of this model on the hold out calibration dataset $\mathcal{D}_2$:

$$R_i = |y_i - \widehat{m}_{n_1}(x_i)|,\ i \in \mathcal{D}_2 \tag{3.8}$$

[4] This method can be found under the name of inductive conformal prediction. We will stick to the machine learning name of split conformal prediction.

[5] It can be any machine learning such as kernel ridge regression, splines, ensembles methods, neural networks...

After that we can apply the finite sample correction on the residuals $R_i,\ i \in \mathcal{D}_2$ to compute the adjusted quantile:

$$\widehat{q}_{n_2} = Q_{\frac{\lceil (1-\alpha)(n_2+1) \rceil}{n_2}} \left(\frac{1}{n_2} \sum_{i=1}^{n_2} \delta_{R_i} \right) \tag{3.9}$$

Then the prediction set writes for $x_{n+1} \in \mathcal{X}$:

$$\begin{aligned} \widehat{C}_{(n_1,n_2)}(x_{n+1}) &= \{y:\ R_{n+1} \leq \widehat{q}_{n_2}\} \\ &= \{y:\ |y_{n+1} - \widehat{m}_{n_1}(x_{n+1})| \leq \widehat{q}_{n_2}\} \\ &= [\widehat{m}_{n_1}(x_{n+1}) - \widehat{q}_{n_2},\ \widehat{m}_{n_1}(x_{n+1}) + \widehat{q}_{n_2}] \end{aligned} \tag{3.10}$$

This procedure gives birth to the following theorem:

Theorem 1 (Marginal coverage for split conformal prediction)
With the only assumption that $(x_1, y_1), \ldots, (x_n, y_n), (x_{n+1}, y_{n+1})$ are exchangeable, the split conformal prediction set satisfies[6]

$$\mathbb{P}\left(y_{n+1} \in \widehat{C}_{(n_1,n_2)}(x_{n+1}) \right) \geq 1 - \alpha. \tag{3.11}$$

[6] The proof is given in Harris Papadopoulos et al. (2002) and Vovk et al. (2005). It is immediate by observing that the R_i are exchangeable.

Remark 1. *Split conformal prediction takes its name from the fact that we split the whole dataset into two. This is mandatory. Indeed let us suppose for a moment that we do not split the dataset and consider only $\mathcal{D}_n$. First, train the regression function on $\mathcal{D}_n$: $\widehat{m}_n : \mathcal{X} \to \mathcal{Y}$. Then compute the residuals $R_i = |y_i - \widehat{m}_n(x_i)|, i \in \mathcal{D}_n$. For a new test point (x_{n+1}, y_{n+1}), we have:*

$$\mathbb{P}\left(y_{n+1} \in \widehat{C}_n(x_{n+1}) \right) \geq 1 - \alpha \iff \mathbb{P}(R_{n+1} \leq \widehat{q}_n) \geq 1 - \alpha \tag{3.12}$$

The last inequality only holds when R_{n+1} is exchangeable with $R_1, \ldots, R_n$. This is not *the case because the residuals $R_i, i \in \mathcal{D}_n$ will tend to be smaller since $\widehat{m}_n$ was trained on $\mathcal{D}_n$, whereas R_{n+1} will be prone to be larger as $\widehat{m}_n$ was not trained on x_{n+1}. Therefore, $R_1, \ldots, R_n, R_{n+1}$ are not exchangeable. Splitting the dataset into training and calibration allows for exchangeable residuals.*

Training conditional coverage is possible for split conformal prediction, and was proven by Vovk (2012a).

Theorem 2 (Train-conditional coverage for split conformal prediction)
With the only assumption that $(x_1, y_1), \ldots, (x_n, y_n), (x_{n+1}, y_{n+1})$ are exchangeable and for any $\delta \in (0, 1)$, the split conformal prediction set satisfies[7]

$$\mathbb{P}[\text{COV}] \geq 1 - \delta \tag{3.13}$$

where COV *is the event of satisfying the coverage guarantee:*

$$\mathbb{P}\left(y_{n+1} \in \widehat{C}_{(n_1,n_2)}(x_{n+1}) | \mathcal{D}_{n_1}, \mathcal{D}_{n_2} \right) \geq 1 - \alpha \tag{3.14}$$

[7] The proof can be found in Vovk (2012a) see proposition 2a and in Bian et al. (2023) see theorem 1.

We illustrate the performances of split conformal prediction in Figure 3.1. More details are available in Appendix B.

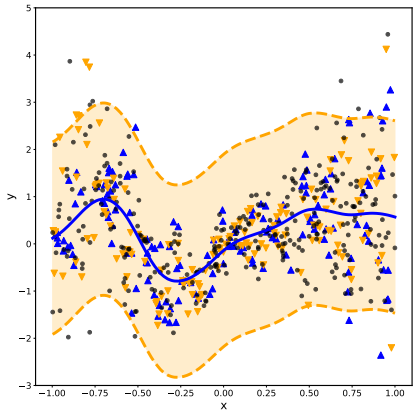


Figure 3.1: Experiment 4 - Split conformal prediction above a Gaussian process model.

Split conformal prediction solves the main problem of full conformal prediction. It is less computationally demanding and achieves training-conditional coverage with minimal assumptions. Unfortunately, splitting the data is not desired at all, especially in Safran future usecases where we will not have a lot of data points. In between full and split conformal prediction we can leverage cross-validation techniques to (hopefully) keep the best of both worlds.

3.3 Jackknife+

Cross-conformal prediction was introduced by Vovk (2012b) and Vovk et al. (2019) and used cross-validation instead of a single split. Unfortunately this method is not that easy to use in practice and is not popular. Moreover, there does not exist a train-conditional result for cross-conformal prediction. Likewise, the jackknife version which makes n splits, where n is the number of observations, does not have any result on coverage (marginal or train-conditional), see Lei et al. (2017). Fortunately, Barber et al. (2020a) introduced a new idea to the jackknife method, which allows for coverage guarantees. Let $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(x_i, y_i) \sim \mathcal{P}$ where $\mathcal{P}$ is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(x_{n+1}, y_{n+1}) \sim \mathcal{P}$ be a test point.
Jackknife+ is very similar to jackknife except that we do not need to compute the point prediction regression function trained on the whole $\mathcal{D}_n$ dataset. Simply train the leave-one-out point predictors:

$$\widehat{m}_{-i} \colon \mathcal{X} \to \mathcal{Y}, \quad \text{trained on } \mathcal{D}_n \backslash \{(x_i, y_i)\} \tag{3.15}$$

and compute the leave-one-out residuals

$$R_i = |y_i - \widehat{m}_{-i}(x_i)|, \ i \in [\![1, n]\!] \tag{3.16}$$

The trick in Jackknife+ is to compute not one but two quantiles and to use the leave-one-out point predictors instead of $\widehat{m}_n$. For the test point $x_{n+1} \in \mathcal{X}$, compute the following:

$$\begin{aligned} E_i^- &:= \widehat{m}_{-i}(x_{n+1}) - R_i, \ i \in [\![1, n]\!] \text{ and} \\ E_i^+ &:= \widehat{m}_{-i}(x_{n+1}) + R_i, \ i \in [\![1, n]\!]. \end{aligned} \tag{3.17}$$

Then, for the lower bound of the prediction set compute the α adjusted quantile of $\left(E_1^-, \ldots E_n^-\right)$

$$\widehat{q_n^-} = Q_{\frac{\lceil \alpha(n+1) \rceil}{n}} \left(\frac{1}{n} \sum_{i=1}^{n} \delta_{E_i^-} \right), \tag{3.18}$$

and for the upper bound compute the $1-\alpha$ adjusted quantile of $\left(E_1^+, \ldots, E_n^+\right)$

$$\widehat{q_n^+} = Q_{\frac{\lceil (1-\alpha)(n+1) \rceil}{n}} \left(\frac{1}{n} \sum_{i=1}^{n} \delta_{E_i^+} \right). \tag{3.19}$$

The prediction set then writes:

$$\widehat{C}_n(x_{n+1}) = \left[\widehat{q_n^-}, \widehat{q_n^+}\right] \tag{3.20}$$

Theorem 3 (Marginal coverage for jackknife+)
If $(x_1, y_1), \ldots, (x_n, y_n), (x_{n+1}, y_{n+1})$ are exchangeable, then the jackknife+ prediction set satisfies[8]*:*

$$\mathbb{P}\left(y_{n+1} \in \widehat{C}_n(x_{n+1})\right) \geq 1 - 2\alpha \tag{3.21}$$

[8] A proof for this theorem is to be found in Barber et al. (2020a) in section 6.

Barber et al. (2020a) provides another theorem on the jackknife+ method that states that the bound $1 - 2\alpha$ is the best the jackknife+ method can achieve. To achieve better bounds one must use additional assumptions. In the same line, Bian et al. (2023) shows that without further assumptions the jackknife+ can not achieve train-conditional coverage. Finally, R. Liang et al. (2024) show training-conditional coverage for the jackknife+ using an algorithm stability assumption. Writing $\beta_{m,n}^{\text{out}}$ the out-of-sample m-stability, as it is defined in R. Liang et al. (2024) (see their definition 3.1). Their theorem writes:

Theorem 4 (Train-conditional coverage for jackknife+)
Suppose the algorithm used to train the point predictor regression functions is symmetric and assume that $Y|X$ has density $f_{Y|X}(y|x)$ with respect to the Lebesgue measure and define

$$B_{dens} = \mathbb{E}\left[\sup_{y\in\mathbb{R}} f_{Y|X}(y|X)\right] \tag{3.22}$$

Then, for any $m \geq 1$ and $\delta \in (0,1)$[9]

$$\mathbb{P}\left[\text{COV}\right] \geq 1 - 3\delta - \sqrt[4]{2B_{dens}\beta^{out}_{m,n-1}} \tag{3.23}$$

where COV *is the event of the training-conditional coverage guarantee,*

$$\mathbb{P}\left(y_{n+1} \in \widehat{C}^{\gamma}_{n}(x_{n+1})|\mathcal{D}_n\right) \geq 1 - \alpha - 3\sqrt{\frac{\log(1/\delta)}{2\min(n,m)}} - 4\sqrt[4]{2B_{dens}\beta^{out}_{m,n-1}}. \tag{3.24}$$

[9] Proof is given in R. Liang et al. (2024).

Finally, Barber et al. (2020a) also presented a Minmax variant and a K-fold cross-validation variant of their Jackknife+ algorithm. Those variants benefit of similar theorems. As an example, Figure 3.2 displays the Jackknife+ method wrapped around a Gaussian process. Again, details are in Appendix B.

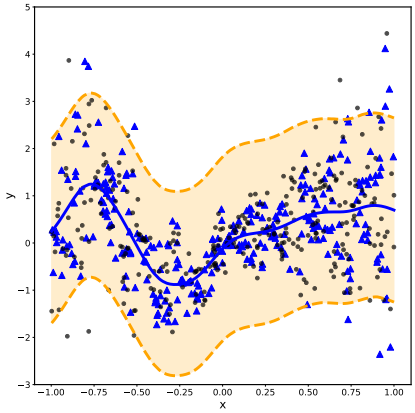


Figure 3.2: Experiment 4 - Jackknife+ conformal model with a Gaussian process.

Remark 2. *Using jackknife and cross-validation can seem computationally inefficient but some machine learning models do not need to be refitted to compute the leave-one-out residuals (linear smoother) and for random forest, the leave-one-out predictors can be computed at low cost.*

With that we cover most of the results in conformal prediction. In the next section we are going to discuss adaptativity of conformal prediction methods.

3.4 Discussion about adaptativity

Adaptativity of the prediction bands was defined as key property for prediction set regression functions. Unfortunately, for all methods discussed above, the prediction set is of constant width[10]. Adaptativity of the prediction set can be defined as the ability to have a width that is a function of x. The question is, is it possible to have adaptative prediction sets and is there a simple way to achieve that in conformal prediction methods?
The answer is yes. For all conformal methods (full, split, cross, jackknife, jackknife+ and variants) we presented them using residuals (absolute value between truth and prediction) to measure how good or bad the point predictor was. This does not need to be the case and we can replace the residuals with a more general score metric of our choice. Most of the methods we are going to present next are "scaled" version of the residuals, and the difference between them lies in how the scale factor is computed, with the exception of conformalized quantile regression that we are going to present last. In the following, we will use the letter S for the scores to replace R for the residuals.

[10] Except for full conformal prediction, but in practice the prediction set is almost of constant width.

3.4.1 Rescaling

Let us suppose now that we have access to an estimation of the standard deviation of the prediction. Instead of computing the residuals, we can

compute a scaled version using the estimation of the standard deviation:

$$S_i = \frac{|y_i - \widehat{m}(x_i)|}{\widehat{\sigma}(x_i)},\ i \in \mathcal{D}_{\text{cal}} \tag{3.25}$$

where $\mathcal{D}_{\text{cal}}$ is the calibration dataset used in each specific conformal method, and $\widehat{\sigma}$ is an estimation of the standard deviation. Then, we leverage once again the finite sample correction on the scores $S_i,\ i \in \mathcal{D}_{\text{cal}}$:

$$\widehat{q}_{\text{cal}} = Q_{\frac{\lceil (1-\alpha)(n_{\text{cal}}+1) \rceil}{n_{\text{cal}}}} \left(\frac{1}{n_{\text{cal}}} \sum_{i \in n_{\text{cal}}} \delta_{S_i} \right). \tag{3.26}$$

Finally, we can compute the adaptative prediction bands:

$$\widehat{C}_n(x_{n+1}) = \left[\widehat{m}(x_{n+1}) \pm \widehat{\sigma}(x_{n+1}) \cdot \widehat{q}_{\text{cal}} \right]. \tag{3.27}$$

The estimation of the standard deviation can be as simple as estimating it on the residuals of the calibration dataset, this is called Studentized residuals and is presented by Lei et al. (2017). Ensemble methods seen in Section 2.3 can be of interest here. Simply compute

$$\widehat{\sigma}_B(x_i) = \sqrt{\frac{1}{B-1} \sum_{b=1}^{B} \left(\widehat{m}_b(x_i) - \widehat{m}_B(x_i) \right)^2},\ i \in \mathcal{D}_{\text{cal}} \tag{3.28}$$

and the prediction bands become adaptative, see Johansson et al. (2014). Finally, we can take advantage of Gaussian processes (Section 2.5), because they estimate a Gaussian distribution per observation, and use the posterior variance[11]:

$$\widehat{\sigma}_{GP}(x_i) = \sqrt{k(x_i, x_i) - k(\boldsymbol{X}, x_i) k(\boldsymbol{X}, \boldsymbol{X}) k(\boldsymbol{X}, x_i)},\ i \in \mathcal{D}_{\text{cal}} \tag{3.29}$$

and the prediction bands become adaptative. For conformalized Gaussian processes see Harris Papadopoulos (2023) for the full and split variants. Jaber et al. (2024) for the Jackknife+ version, and Pion et al. (2024) for asymmetric scores. In order to illustrate those models we chose the J+GP model developed by Jaber et al. (2024) in Figure 3.3. Details are in Appendix B.

[11] This method is presented using $\max(\widehat{\sigma}(x_i), \varepsilon),\ \varepsilon > 0$ in order to tackle the case where the Gaussian process model interpolates.

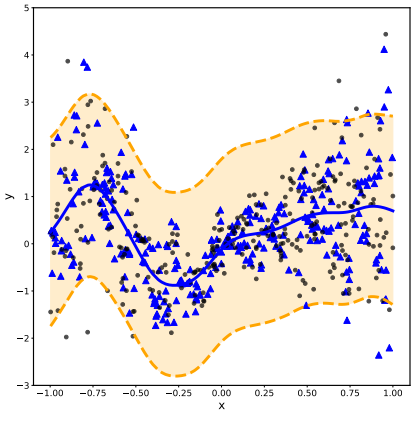


Figure 3.3: Experiment 4 - J+GP model.

3.4.2 Conformalized quantile regression

The idea of conformalized quantile regression (Romano et al. (2019)) is slightly different from what we have discussed so far. Instead of estimating the mean function of the data, conformalized quantile regression leverages quantile regression models, discussed in Section 2.4, to compute the conformal scores.

For the split conformal quantile regression the method goes as follow. Let $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(x_i, y_i) \sim P$ where P is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(x_{n+1}, y_{n+1}) \sim P$ be a test point. We divide the training dataset into two, a proper training dataset $\mathcal{D}_{n_1}$ and $\mathcal{D}_{n_2}$ a calibration dataset. Using an algorithm of choice for quantile regression, we then estimate the quantile regression function at two quantiles, α_{lo} and α_{hi} such that $\alpha_{\text{lo}} + \alpha_{\text{hi}} = \alpha$, the targeted error rate:

$$\widehat{q}_{\alpha_{\text{lo}}} : \mathcal{X} \to \mathcal{Y},\ \text{estimation of the } \alpha_{\text{lo}} \text{ quantile function trained on } \mathcal{D}_{n_1}.$$
$$\widehat{q}_{\alpha_{\text{hi}}} : \mathcal{X} \to \mathcal{Y},\ \text{estimation of the } \alpha_{\text{hi}} \text{ quantile function trained on } \mathcal{D}_{n_1}.$$

The scores are then computed in the following manner:

$$S_i = \max \left\{ \widehat{q}_{\alpha_{\text{lo}}}(x_i) - y_i;\ y_i - \widehat{q}_{\alpha_{\text{hi}}}(x_i) \right\},\ i \in \mathcal{D}_{n_2}. \tag{3.30}$$

Next, we compute the finite sample correction quantile on the scores to obtain $\widehat{q}_{n_2}$, and the prediction set writes:

$$\widehat{C}_{(n_1,n_2)}(x_{n+1}) = [\widehat{q}_{\alpha_{\text{lo}}}(x_{n+1}) - \widehat{q}_{n_2};\ \widehat{q}_{\alpha_{\text{hi}}}(x_{n+1}) + \widehat{q}_{n_2}]\,. \tag{3.31}$$

Romano et al. (2019) proves the following theorem:

Theorem 5 (Marginal coverage for Conformalized Quantile Regression) *If* $(x_i, y_i), i = 1\ldots, n, n+1$ *are exchangeable, the prediction interval constructed using split CQR satisfies:*

$$\mathbb{P}\left(y_{n+1} \in \widehat{C}_{(n_1,n_2)}(x_{n+1})\right) \geq 1 - \alpha. \tag{3.32}$$

The performances of CQR are showcased in Figure 3.4, where we used the same quantile regression model as in Figure 2.2. Details are in Appendix B. What is not stated in the original paper and would be interesting to investigate is, i) can CQR be adapted to the jackknife+ algorithm, ii) can CQR achieve train-conditional coverage and under which assumptions?

Conformal prediction establishes itself as the only framework to build distribution free prediction bands with coverage guarantees. The exchangeability assumption is weak yet very specific. This means that conformal prediction is difficult to extend to non exchangeable settings. Moreover, the solutions for adaptative bands are not sufficient. The bands still fails to adapt in some scenarios. We will now present in the next three chapters a new paradigm to construct distribution free prediction bands.

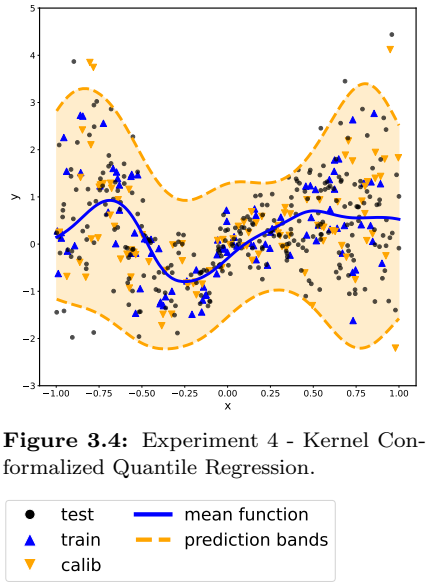


Figure 3.4: Experiment 4 - Kernel Conformalized Quantile Regression.

Chapter 4

Kernel methods to assess uncertainty in the predictions

As mentioned in the previous chapter, uncertainty quantification is a vast part of machine learning, with a lot of progress to be made. Here, we investigate a radically different and new approach to the question. This method was proposed by T. Liang (2022) and relies on kernels and a new representer theorem for non-negative functions from Marteau-Ferey et al. (2020). Compared to conformal prediction, this new method seems to be more adaptive to the data and the coverage guarantee is conditional to the training set with minimal assumptions. In this chapter we will first present the model proposed in the original paper of T. Liang (2022), then we will study a related method proposed later by Fan et al. (2024).

4.1 Minimising the nuclear norm

4.1.1 Introduction to the semi-definite problem

Following the paper of Marteau-Ferey et al. (2020), Liang used the representer theorem to develop a model to estimate the variance function. The approach involves estimating this function in a Reproducible Kernel Hilbert Space (RKHS) where all the functions would be non-negative, because the variance is a non-negative function. We will dive into more details on this in the Section 5.1. The variance is thus modeled as the following non negative function, for any $x \in \mathcal{X}$,

$$v(x) = \phi(x)^\top \mathcal{A}\phi(x) = \langle \phi(x), \mathcal{A}\phi(x) \rangle_{\mathcal{H}} \geq 0$$

where $\mathcal{A}$ is a *positive semi-definite* operator from the RKHS $\mathcal{H}$ to itself. What Liang proposes is to find the best suitable operator $\mathcal{A}$ to estimate the variance function.

Let us consider $\mathcal{H}$ the RKHS in which we are going to estimate the variance function[12] and let $\widehat{m}_p$ be an estimator of the mean function[13]. What Liang proposes to estimate prediction bands around this mean estimator is to solve the following semi-definite programming (SDP) problem:

Problem 1. *Let $\widehat{m}_p$ be an estimator of the mean function and $\mathcal{D}_n$ be the*

[12] Liang also suggests another version that estimates the mean function and the variance function at the same time. We will touch on that later, see Chapter 6.

[13] This estimator is supposed to have been trained on a completely different dataset. We denote $\mathcal{D}_p$ the dataset used to train this mean estimator, where p stands for pre-trained.

training dataset for the prediction bands.

$$\min_{\mathcal{A}:\mathcal{H}\to\mathcal{H}} \ ||\mathcal{A}||_\star \tag{4.1a}$$

$$s.t. \ \langle \phi_{x_i}, \mathcal{A}(\phi_{x_i})\rangle_{\mathcal{H}} \geq \left(y_i - \widehat{m}_p(x_i)\right)^2, \ i \in [\![1, n]\!] \tag{4.1b}$$

$$\mathcal{A} \succeq 0 \tag{4.1c}$$

Problem 1 looks for the positive semi-definite (PSD) operator that has minimal operator norm[14] and such that, on all training data points, the estimated variance is at least greater than the residuals between the true output value and the mean estimation. This constraint, Equation (4.1b), is interesting to study, since it implies that the prediction bands cover all training data points. Therefore we will have a 100% coverage on the training data points. The prediction bands are going to be supported by a few data points, that are the ones with the greatest residuals[15]. If one can solve such a problem, Liang proposes to construct the prediction bands for a new data point $x \in \mathcal{X}$ as:

$$\widehat{C}_{(p,n)}(x, \delta) = \left[\widehat{m}_p(x) \pm \sqrt{(1+\delta)\cdot\langle \phi_x, \widehat{\mathcal{A}}_n \phi_x\rangle}\right] \tag{4.2}$$

where $\delta > 0$ is a constant for now. Unfortunately, Problem 1 is infinite dimensional, but using results from Marteau-Ferey et al. (2020) on operators we can write the following theorem:

Theorem 6 (Representation form of Liang)
Let $\widehat{m}_p$ be an estimator of the mean function and $\mathcal{D}_n$ the training dataset. Denote k the kernel associated to the RKHS $\mathcal{H}$ defined in Problem 1 and $\boldsymbol{K}$ the kernel matrix constructed from k and $\mathcal{D}_n$. Using Theorem 1 *from Marteau-Ferey et al. (2020), Problem 1 can be written:*

$$\min_{\boldsymbol{B}\in\mathbb{S}^n} \ \mathrm{Tr}(\boldsymbol{K}\boldsymbol{B}) \tag{4.3a}$$

$$s.t. \ \langle \boldsymbol{K}_{x_i}, \boldsymbol{B}\boldsymbol{K}_{x_i}\rangle \geq \left(y_i - \widehat{m}_p(x_i)\right)^2, \ i \in [\![1, n]\!] \tag{4.3b}$$

$$\boldsymbol{B} \succeq 0 \tag{4.3c}$$

Let $\delta > 0$. Using the solution of this problem, we can write the prediction bands at $x \in \mathcal{X}$:

$$\widehat{C}_{(p,n)}(x, \delta) = \left[\widehat{m}_p(x) \pm \sqrt{(1+\delta)\cdot\langle \boldsymbol{K}_x, \widehat{\boldsymbol{B}}_n \boldsymbol{K}_x\rangle}\right]. \tag{4.4}$$

Liang proposes a way of choosing the δ value from the data in a way that ensures the coverage guarantee to be at a user specified level α. Before delving into that procedure, let us study the following theorem, that ensures the training-conditional coverage guarantee obtained from Equation (4.4). The kernel for the variance function must satisfy the following assumptions in order to verify the guarantee.

A1 The variance kernel satisfy $\sup_{x\in\mathcal{X}} K(x,x) < C$ and its associated integral operator $\mathcal{T}\colon \mathcal{H} \to \mathcal{H}$ satisfy $\lambda_j(\mathcal{T}) \leq Cj^{-\tau}, j \in \mathbb{N}$, for some constant τ.

A2 There exist constants $\eta \in (0,1)$, $\xi > 0$ such that $\mathbb{P}\left(y^2 > \xi \cdot K(x,x) | x = x\right) > \theta$ holds for all $x \in \mathcal{X}$.

A3 There exist a constant $\omega > 0$ such that $\mathbb{P}\left(y^2 > t \cdot K(x,x)\right) < \exp(-Ct^\omega)$ for all $t \geq 1$.

[14] Note that the operator norm for a positive semi-definite trace class operator is equal to its trace: $||\mathcal{A}||_* = \mathrm{Tr}(\mathcal{A})$

[15] Support vector prediction bands

The first assumption $A1$ is similar to asking for the trace integral operator to be bounded. Moreover, the assumption that the dissimilarities between observations is bounded is also something reasonable to make. The assumptions $A2$ and $A3$ are similar, in the sense that they require for the uncertainty to be non-trivial (that it exists) and to be non-wild, decaying in an exponential fashion.

Theorem 7 (Liang coverage guarantee)
Suppose we used the dataset $\mathcal{D}_p$ to train the mean estimator $\widehat{m}_p$. Consider $\mathcal{D}_n$ the training data, and $(x, y) \sim \mathcal{P}$ a fresh data point. Denote by $\widehat{Opt}_n$ the objective value of the Problem 1 and by $\widehat{C}_{(p,n)}(x, \delta)$ the prediction band as defined in Equation (4.4). Under the assumptions A1-3 for the kernel, for any $\delta \in (0, 1]$ the following train-conditional coverage guarantee holds with probability at least $1 - n^{-10}$:

$$\mathbb{P}\left(y \in \widehat{C}_{(p,n)}(x, \delta) \mid \boldsymbol{\mathcal{D}_n}\right) \geq 1 - \frac{1}{\delta} \cdot (\widehat{Opt}_n \vee 1) \cdot \sqrt{\frac{C_{\tau,\xi,\eta,\omega} \cdot \log(n)}{n}} \quad (4.5)$$

$$\text{and } \widehat{Opt}_n \leq (\log(n))^{c_\tau} \quad (4.6)$$

where $C_{\tau,\xi,\eta,\omega}$ and c_τ are constants that depends on parameters from the assumptions A1-3.

A proof for this theorem is the core of the article by T. Liang (2022). A few comments must be made on this theorem. First, the achieved coverage is train-conditional. Second, it holds in finite sample. Third, the prediction bands are, at least theoretically adaptive because the variance function is a function of x, see Equation (4.4). However, this does not allow for a user-specified error level, unlike conformal prediction.

To allow for such a coverage control in his model, Liang proposes a calibration procedure using a second dataset $\mathcal{D}_m$. Algorithm 1 presents the procedure, while Theorem 8 ensures the theoretical validity of this algorithm.

Theorem 8 (Liang's calibration)
Suppose we used the dataset $\mathcal{D}_p$ to train the mean estimator $\widehat{m}_p$. Let $\alpha \in (0, 1)$ be the coverage level. Consider $\mathcal{D}_n$, $\mathcal{D}_m$ a training and a calibration datasets, and $(x, y) \sim \mathcal{P}$ a fresh data point. Suppose that there exists large enough constants $\Delta > 0$ and $L > 0$ such that:

$$i)\, \mathbb{P}\left(y \in \widehat{C}(x, \Delta)\right) = 1, \quad ii)\, \left|\frac{d}{d\delta}\mathbb{P}\left(y \notin \widehat{C}(x, \delta)\right)\right| < L. \quad (4.7)$$

For a calibration size m large enough such that

$$\sqrt{\frac{\log\left(\lceil \log_2\left(\frac{2L(\Delta+1)}{\alpha}\right)\rceil + 1\right) + 10\log(m)}{m}} \leq \frac{1}{4}\alpha\,,$$

the value $\widehat{\delta}_\alpha$, computed as per Algorithm 1, satisfies the coverage control:

$$\mathbb{P}\left(y \in \widehat{C}_{(p,n,m)}(x, \widehat{\delta}_\alpha) \mid \mathcal{D}_n, \mathcal{D}_m\right) \geq 1 - \alpha \quad (4.8)$$

with probability at least $1 - 2m^{-10}$.

From a theoretical point of view this calibration procedure is strong, as it proves that Liang's model can achieve a user specified coverage level. On a more practical perspective, this procedure is not very efficient. First of all, the hypotheses in Equation (4.7) are really strong. Especially the

Algorithm 1: Calibration for Coverage Control in T. Liang (2022)

Data: Calibration set $\mathcal{D}_m$, coverage level $1-\alpha$;
Result: Calibrated $\widehat{\delta}_\alpha$;
Initialize $\delta = -1$;
while $\frac{1}{m}\sum_{j=1}^{m} \mathbb{1}_{\left[y'_j \notin \widehat{C}(x'_j,\delta)\right]} > \frac{3}{4}\alpha$, **do**
  $\delta \leftarrow \frac{\delta+\Delta}{2}$;
end
return $\widehat{\delta}_\alpha \leftarrow \delta$.

Fig	$\widehat{\delta}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 4.1a	1.0	1.0	0.89	0.83	7.9
Figure 4.1b	1.0	1.0	0.91	0.91	9.5
Figure 4.1c	0.875	1.0	0.97	0.96	10.9
Figure 4.1d	0.5	1.0	0.97	0.97	11.0

Table 4.1: Statistics about experiments in Figure 4.1.

first one forces the true variation of the data to be of finite support. A normal error rate is thus prohibited. Second, this procedure essentially replaces the hyperparameter δ by another hyperparameter Δ. In practice the choice of Δ is very hard. If Δ is chosen too small, we will get stuck in the while loop. If Δ is too big, then the algorithm will land on a $\widehat{\delta}_\alpha$ that is too big and the prediction bands will become uninformative.

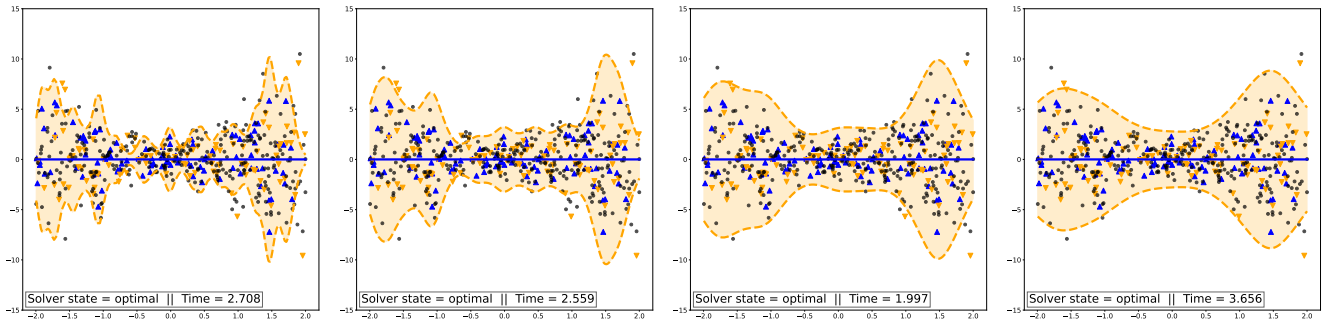
In summary, T. Liang (2022) proposed a brand new way of computing prediction band that is at complete odds with previous methods. His method guarantees a train-conditional coverage, is non asymptotic and data adaptive. However, his proposed calibration procedure to obtain a coverage at user specified level is wobbly. Moreover this model has numerous hyperparameters to setup: the length scale for the variance kernels, and Δ for the calibration procedure. But unfortunately there exist no way to choose their value in the original paper.

4.1.2 Empirical investigation

The model proposed by T. Liang (2022) has three hyperparameters which impact the results considerably. The first one is the kernel length scale. The second one is Δ in the calibration procedure. Finally, the third one is the estimation of the mean function $\widehat{m}_p$.

In the following illustrations, we fix the mean estimator to a Gaussian process trained on a separate dataset. Thus, we only explore two hyperparameters, the length scale of $\boldsymbol{K}$ and the calibration parameter Δ. Associated with each figure, we provide a table with numerous statistics such as the value of the calibrated $\widehat{\delta}_\alpha$, the coverage on each datasets, training, calibration and test, and the average length of the prediction bands on the test dataset.

Experiment 2. The first thing to note is that the Gaussian process seems to learn correctly the null mean function. We present hereafter two figures, for two different values of Δ. Figure 4.1 showcases the value $\Delta = 1$. In Table 4.1 we present all the metrics of those four figures. First of all, we can observe the expected 100% coverage on the training data. However,



(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

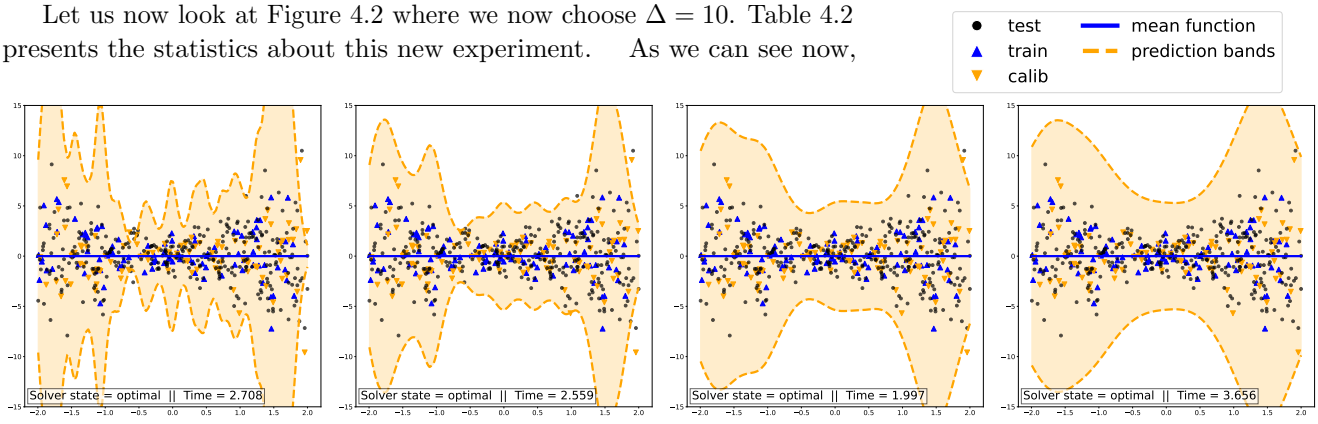
Figure 4.1: Experiment 2 - Liang B - $\Delta = 1$ - Evolution of the prediction bands shape with regard to the length scale.

Fig	$\widehat{\delta}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 4.2a	10.0	1.0	0.96	0.96	18.6
Figure 4.2b	4.5	1.0	0.97	0.98	15.8
Figure 4.2c	4.5	1.0	0.98	0.99	18.8
Figure 4.2d	4.5	1.0	0.1.0	1.0	21.0

Table 4.2: Statistics about experiments in Figure 4.2.

only for the length scales 0.4 and 0.6 we obtain the desired coverage of 0.95 on the test data. This suggests that the calibration procedure is sub optimal. Indeed for the length scales that did not attained the desired coverage level we have $\Delta = \widehat{\delta}_\alpha$. Our understanding is that Δ was too small to encompass 0.95% of the calibration data. Unfortunately, this behaviour depends on the training data, the calibration data, the length scale and is therefore not predictible. Regarding the differences between the length scales, we can see that a smaller length scale leads to more "wiggly" bands, while higher length scales lead to smoother bands. Overall, the shape of the bands seems to fit quite nicely the data, and the adaptivity is strong. In the middle of the figures the bands are quite narrow, while on the edges the bands are really wide. Adaptivity seems really embedded in the learning of the operator. This is really contrasting with conformal prediction where adaptivity is harder to achieve. Sadly, the tuning of the hyperparameters are so hard that this method seems impossible to use in practice.

Let us now look at Figure 4.2 where we now choose $\Delta = 10$. Table 4.2 presents the statistics about this new experiment. As we can see now,



(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

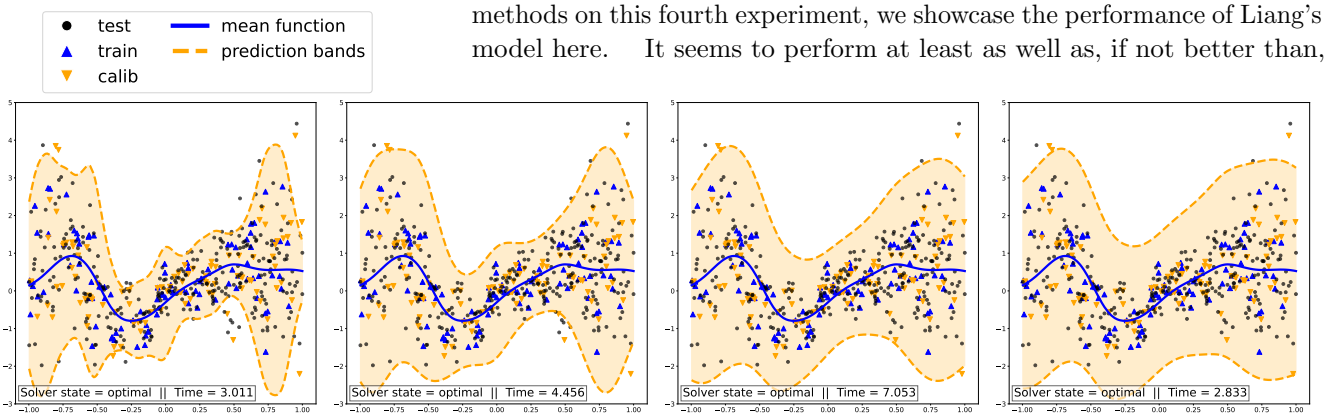
Figure 4.2: Experiment 2 - Liang B - $\Delta = 10$ - Evolution of the prediction bands' shape with regard to the length scale.

the bands are a lot wider across all length scales. Indeed we manage to achieve the desired coverage on the test sample each time, but at what cost? In fact, the average length for all length scales skyrocketed, which is

not desirable. Again, this sheds the light on the calibration procedure that is not optimal. For length scales 0.2 and above, $\widehat{\delta}_\alpha = 4.5$. The calibration procedure stopped after one iteration because Δ was chosen too big. This procedure is not reliable and impossible to tune. The hyperparameter Δ is not connected to the data in a meaningful way, which makes it impossible to choose or to interpret.

We believe this example is enough to showcase the weakness of Liang calibration procedure. Before studying the paper by Fan et al. (2024), let us take a look at the performance of Liang on experiment four, as it is the only one without a null mean function.

Experiment 4. For visual reasons, and because we tested all conformal methods on this fourth experiment, we showcase the performance of Liang's model here. It seems to perform at least as well as, if not better than, some conformal prediction models, but with the improved adaptivity. Moreover, the flexibility of choosing the length scale looks promising, as we can have very different prediction bands. We do not develop on the impact of Δ as we clearly understood that the calibration procedure was flawed.



(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

Figure 4.3: Experiment 4 - Liang B - $\Delta = 1$ - Evolution of the prediction bands' with regard to the length scale.

4.2 Minimising the average width

4.2.1 General setting

Fan et al. (2024) proposed a new model, based on a different idea from Liang's, but of which a special case resembles Liang's model. The idea here is to find an estimate of the variance function within a set of functions $\mathcal{V}$ with the smallest average length.

We will see that the key difference compared to Liang's model lie in the objective function. But first, let $\widehat{m}_p$ be an estimator of the mean function trained on the dataset $\mathcal{D}_p$ and consider $\mathcal{V}$ a set of candidates for the estimation of the prediction bands. Given a training dataset $\mathcal{D}_n$, the following problem looks for the thinnest prediction bands possible, among the set of candidates:

$$\min_{v \in \mathcal{V}} \ \frac{1}{n} \sum_{i=1}^{n} v(x_i) \tag{4.9a}$$

$$\text{s.t.} \ \ v(x_i) \geq \left(y_i - \widehat{m}_p(x_i)\right), \ i \in [\![1, n]\!]. \tag{4.9b}$$

Similarly to Liang, the constraint in Equation (4.9b) implies that the prediction bands cover the whole training dataset (achieving 100% train-coverage). Therefore, Equation (4.9a) looks for the estimator $\widehat{v}$ with the

smallest average width, among those who achieve 100% train-coverage. We can see that v represents both the variance function and the width of the prediction bands at a given point. By solving Equation (4.9a), Fan suggests a slight modification, in comparison to Liang, to construct the prediction band:

$$\widehat{C}_{(p,n)}\left(x, \delta, \lambda\right) = \left[\widehat{m}_p(x) \pm \sqrt{\lambda\left(\widehat{v}_n(x) + \delta\right)}\right] \tag{4.10}$$

where δ represents a small correction that appears in the lower bound of the coverage guarantee theorem and λ is a scale factor that will be tuned during calibration, in order to achieve coverage at user-specified level.

Next we present the theorem that guarantees the empirical coverage. It relies on a continuity assumption near the best estimator within the set. Let $v^\star$ be the best approximator of v_0 (the true variance function) in $\mathcal{V}$. The assumption writes:

$$\exists D > 0 \ / \ \forall d \in [0, D],\ v^\star + d \in \mathcal{V}. \tag{4.11}$$

Theorem 9 (Fan guarantee coverage)
Let $\widehat{m}_p$ be a mean estimator trained on dataset $\mathcal{D}_p$. Consider a training dataset $\mathcal{D}_n$ and $(x, y) \sim \mathcal{P}$ a fresh data point. Let $\widehat{v}_n$ be the solutions of Equation (4.9a) *and $\widehat{C}_{(p,n)}\left(x, \delta, \lambda\right)$ the resulting prediction bands. If $\mathcal{V}$ is a non-VC class*[16] *of functions that verify Equation* (4.11)*, then for any $\delta, t > 0$, holds the following guarantee with probability at least $1 - e^{-t}$:*

$$\mathbb{P}\left(y \in \widehat{C}_{(p,n)}(x, \delta, \lambda) \ \middle| \ \boldsymbol{\mathcal{D}_n}\right) \geq 1 - \frac{2}{\delta} \cdot \mathcal{R}_n\left(\mathcal{V}\right) - \sqrt{\frac{2t}{n}}. \tag{4.12}$$

[16] This refers to the statistical learning theory. It appears here because the complexity of the function class appears in the bound.

Just like in Theorem 7, the coverage guarantee is training conditional and in finite sample. Once again, the coverage level here is not chosen by the user. In order to achieve the desired coverage level a calibration step is needed and Fan et al. (2024) proposes a different procedure than T. Liang (2022). Their view is that with prediction bands of a 100% coverage on the training data, attaining a specified coverage level is possible through a single scaling of the bands. The calibration procedure is thus learning a scale factor $\widehat{\lambda}_\alpha$ to adjust the prediction bands in Equation (4.10), on a different calibration dataset $\mathcal{D}_m$. Note that we do not use the full

Algorithm 2: Calibration for Coverage Control in Fan et al. (2024)

Input: Calibration set $\mathcal{D}_m$, coverage level $1 - \alpha$, $\widehat{v}_n$ the solutions of Equation (4.9a) on $\mathcal{D}_n$ and $\widehat{C}_{(p,n)}$ as in Equation (4.10);

Output: Calibrated $\widehat{\lambda}_\alpha$;

Compute the $(1-\alpha)^{\text{th}}$ empirical quantile of $\frac{\left(y_i' - \widehat{m}_p(x_i')\right)^2}{\left(\widehat{v}_n(x_i') + \delta\right)}$ on the subset $\mathcal{I}_2$ which is all the calibration data points that are inside the prediction bands before calibration.:

$$\widehat{\lambda}_\alpha \coloneqq \inf_{\lambda}\left\{\lambda \colon \frac{1}{m}\sum_{j=1}^{m} \mathbb{1}_{\left[\left(y_i' - \widehat{m}_p(x_i')\right)^2 > \lambda\left(\widehat{v}_n(x_i') + \delta\right)\right]} \leq \alpha\right\} \tag{4.13}$$

return $\widehat{\lambda}_\alpha$.

calibration dataset here, but only a subset of it. The used subset consists of all the calibration points that falls into the bands before the calibration (i.e. with $\lambda = 1$.). The justification in Fan et al. (2024) for this subsetting

is that if the whole calibration dataset was used, sometimes in their own experimentation they would come across too large values of λ[17]. By using this trick, they constrain the value of λ in $[0, 1]$. Using this calibration procedure the prediction bands are then computed as:

$$\widehat{C}_{(p,n,m)}(x, \delta) = \left[\widehat{m}_p(x) \pm \sqrt{\widehat{\lambda}_{m,\alpha} \cdot (\widehat{v}_n(x) + \delta)}\right]. \tag{4.14}$$

This calibration procedure and those calibrated prediction bands are presented along side the following theorem in the paper by Fan et al. (2024):

Theorem 10
Let $\widehat{C}_{(p,n,m)}(x, \delta)$ be the prediction bands computed using model from Equation (4.9a) and calibration procedure from Algorithm 2. If $\frac{(y-\widehat{m}(x))}{\widehat{v}(x)}$ has a continuous distribution, then with probability at least $1 - 2m^{-10}$, the following coverage holds:

$$\left|\mathbb{P}\left(y \in \widehat{C}_{(p,n,m)}(x, \delta) \mid \mathcal{D}_n, \mathcal{D}_m\right) - (1 - \alpha)\right| \leq 2(1 - p_n) + c \cdot \sqrt{\frac{\log(n)}{m}} \tag{4.15}$$

for some absolute constant c and large enough n and m and with $p_n = \mathbb{P}\left(y \in \widehat{C}_{(p,n,m)}(x, \delta) \mid \mathcal{D}_n\right)$.

This theorem completes the main ideas to quantify uncertainty in Fan et al. (2024). The connection to Liang's model resides in the choice of the set of candidates $\mathcal{V}$.

4.2.2 A special case with a RKHS

The authors proposed numerous ways to choose the set of candidates $\mathcal{V}$, one of them being to consider a Reproducing Kernel Hilbert Space. Thus $\mathcal{V}$ is defined as:

$$\mathcal{V} = \{v(x) = \langle \phi(x), \mathcal{A}\phi(x)\rangle_{\mathcal{H}},\ ||\mathcal{A}||_* \leq r, \mathcal{A} \succeq 0\} \tag{4.16}$$

This transforms problem from Equation (4.9a) into the following problem, that resembles a lot formulation from Equation (4.1a).

Problem 2.

$$\min_{\mathcal{A}:\mathcal{H}\to\mathcal{H}} \ \frac{1}{n}\sum_{i=1}^{n}\langle \phi_{x_i}, \mathcal{A}\phi_{x_i}\rangle_{\mathcal{H}} \tag{4.17a}$$

$$s.t. \ \langle \phi_{x_i}, \mathcal{A}(\phi_{x_i})\rangle_{\mathcal{H}} \geq (y_i - \widehat{m}_p(x_i))^2,\ i \in [\![1, n]\!] \tag{4.17b}$$

$$||\mathcal{A}||_* \leq r \tag{4.17c}$$

$$\mathcal{A} \succeq 0 \tag{4.17d}$$

This problem is still an infinite dimensional problem and we need to leverage Marteau-Ferey et al. (2020) theorems to make it tractable:

Theorem 11 (Representation form of Fan)
Let $\widehat{m}_p$ be any estimator of the mean function trained on the dataset $\mathcal{D}_p$. Using Theorem 4 from Marteau-Ferey et al. (2020), Problem 2 can be written:

$$\min_{\boldsymbol{B}\in\mathbb{S}^n} \ \mathrm{Tr}(\boldsymbol{K}\boldsymbol{B}\boldsymbol{K}) \tag{4.18a}$$

$$s.t. \ \langle \boldsymbol{K}_{x_i}, \boldsymbol{B}\boldsymbol{K}_{x_i}\rangle \geq (y_i - \widehat{m}_p(x_i))^2,\ i \in [\![1, n]\!] \tag{4.18b}$$

$$\mathrm{Tr}(\boldsymbol{K}\boldsymbol{B}) \leq r \tag{4.18c}$$

$$B \succeq 0. \tag{4.18d}$$

[17] Note that we do not observe this phenomenon and will revisit this procedure later on.

Fig	$\widehat{\lambda}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 4.4a	0.87	0.60	0.58	0.51	3.8
Figure 4.4b	0.86	0.77	0.75	0.65	5.3
Figure 4.4c	0.75	0.78	0.83	0.70	5.7
Figure 4.4d	0.62	0.84	0.84	0.77	5.9

Table 4.3: Statistics about experiments in Figure 4.4.

Let $\delta > 0$. Using the solution of this problem and the calibration procedure from Algorithm 2 we can write the prediction bands at $x \in \mathcal{X}$:

$$\widehat{C}_{(p,n,m)}(x, \delta) = \left[\widehat{m}_p(x) \pm \sqrt{\widehat{\lambda}_{m,\alpha} \cdot \left(\langle \boldsymbol{K}_x, \widehat{\boldsymbol{B}} \boldsymbol{K}_x \rangle + \delta \right)} \right]. \tag{4.19}$$

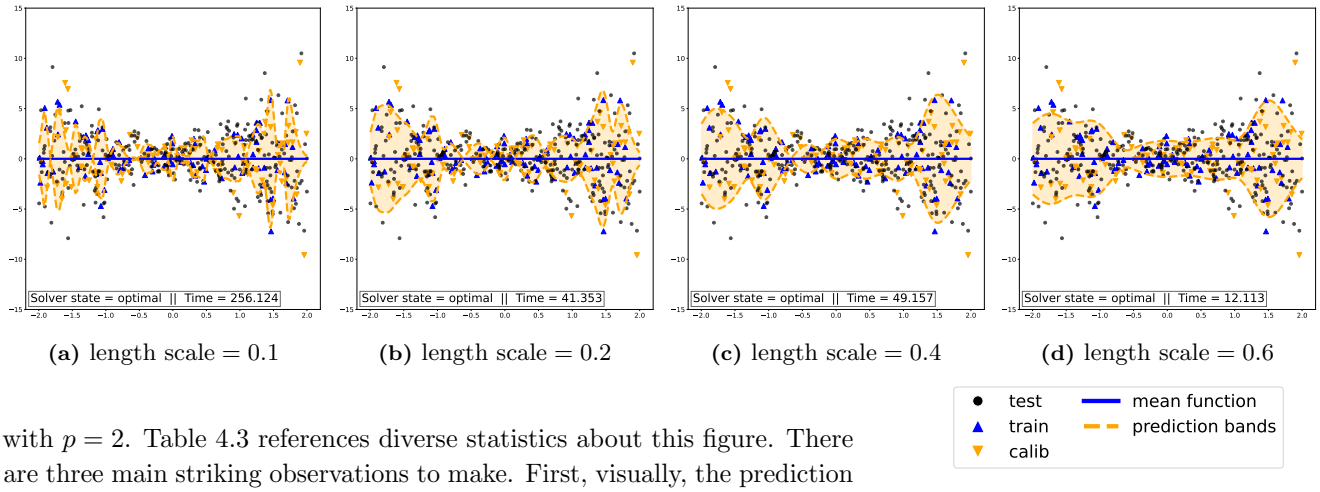
Theorems 9 and 10 apply to Equation (4.18a), and therefore Fan's model quantify uncertainty in a very similar fashion to Liang. The main difference is that its objective is to minimize the average width, furthermore, the operator norm is constrained implicitly rather than explicitly.

4.2.3 Illustration

Now, problem from Equation (4.18a) has four hyperparameters: the kernel length scale, the value of r, the mean prediction function $\widehat{m}_p$ and δ (none for the calibration procedure). As before, we fix the mean predictor to a Gaussian process trained on a separate dataset. We also fix the value of δ to 0.001 for now. The value of r is difficult to choose. Looking at the similarities with Liang's model, we define the following procedure: because Liang aims at minimizing the operator norm, it is unlikely that Fan's model will find a variance function with a smaller operator norm (because it is not trying to). Therefore, if we write $\widehat{\text{Obj}}_{\text{Liang}}$ the value of the operator norm after Liang's optimisation on the exact same problem, we define $r = p \times \widehat{\text{Obj}}_{\text{Liang}}$ with p an integer. We therefore study the impact of p rather than r.

Experiment 2. For this experiment we use the exact same Gaussian process model as the one we used in Section 4.1.2. We also use the same datasets for optimisation and calibration between Liang's and Fan's empirical study. Figure 4.4 shows the prediction bands for Fan's optimisation with $p = 2$. Table 4.3 references diverse statistics about this figure. There are three main striking observations to make. First, visually, the prediction bands shape are a lot more wiggly than Liang's. It appears that because the objective is to minimise the average length, the bands are pinching a

Figure 4.4: Experiment 2 - Fan B - $\delta = 0.001$; $p = 2$ - Evolution of the prediction bands with regard to the length scale.



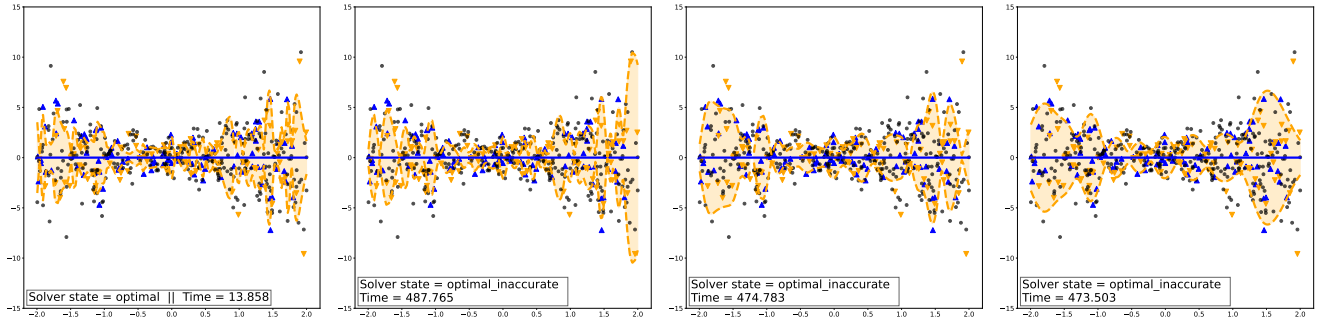
(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

Fig	$\widehat{\lambda}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 4.5a	0.87	0.60	0.58	0.51	3.8
Figure 4.5b	0.86	0.77	0.75	0.65	5.3
Figure 4.5c	0.75	0.78	0.83	0.70	5.7
Figure 4.5d	0.62	0.84	0.84	0.77	5.9

Table 4.4: Statistics about experiments in Figure 4.5.

lot. Second, the train coverage is never at 100%, not even at the targeted level α. This is quite surprising since having a 100% train coverage is within the problem constraints. This is due to the calibration and it is our third remark: as expected the scale coefficient $\widehat{\lambda}_\alpha$ is always strictly inferior to 1. This is the consequence of using a subset of the calibration dataset, and we believe this is what harms Fan's version.

Let us take a look at a different value for p, setting $p = 100$ in Figure 4.5. The statistics are presented in Table 4.4. What we observe is that the



(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

Figure 4.5: Experiment 2 - Fan B - $\delta = 0.001$; $p = 100$ - Evolution of the prediction bands' shape with regard to the length scale.

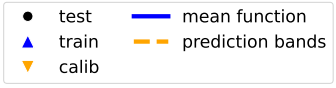
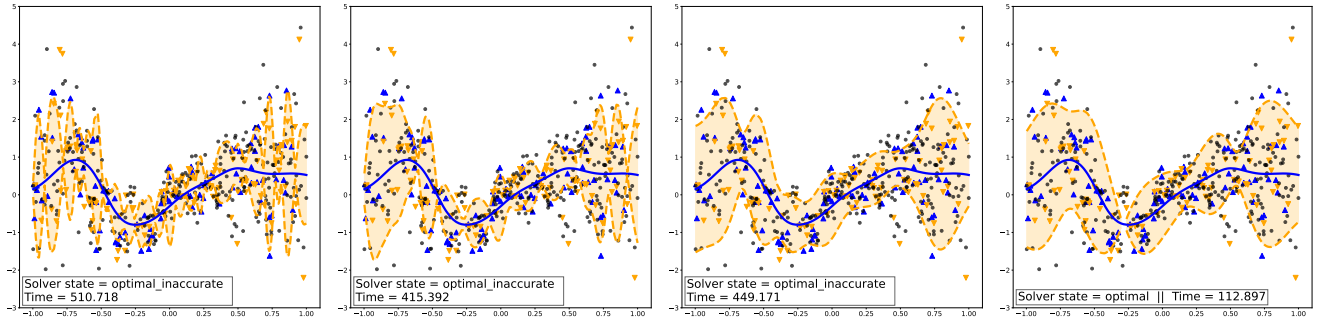


Figure 4.6: Experiment 4 - Fan B - $\delta = 0.001$; $p = 100$ - Evolution of the prediction bands' shape with regard to the length scale.

prediction bands are even more wiggly than with $p = 2$. The coverage statistics are not great either. The influence of p is quite hard to distinguish here. However, what we can note is that the optimisation time is high for Fan's version. More than 400 seconds for a single training, whereas it was only a handful seconds in Liang's case.

Experiment 4. As for Liang, we can compare the performances of Fan's model on the fourth experiment. Compared to Figure 4.3, the prediction

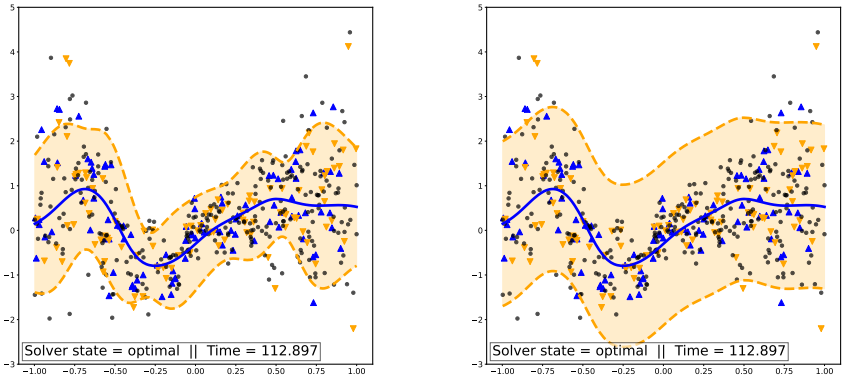


(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

bands are in Figure 4.6, again, oscillating a lot more. Overall the performance is not good, but we believe this is due to a bad calibration procedure. Despite that, the model seems to be able to find good prediction bands

shapes.

Remark 3 (On the impact of δ.)**.** *We are going to simply do a visual investigation of the impact of δ on the shape of the prediction bands. For reference we are going to take the problem with a length scale equal to* 0.6 *and p equal to* 100*. Figure 4.6d has $\delta = 0.001$. In the following Figures 4.7a and 4.7b we are going to consider, respectively, $\delta = 1$ and $\delta = 100$.*



(a) $\delta = 1$ **(b)** $\delta = 100$

Figure 4.7: Experiment 4 - Fan B - $ls = 0.6$; $p = 100$ - Evolution of the prediction bands with regard to δ.

The effect of this hyperparameter is to level the width of the bands. Taking a look at Equation (4.19)*, we see that if δ is much bigger than any values of the variance function, its going to outshine it and δ is a constant. Therefore the prediction bands will loose their adaptivity and tend towards being of constant width.*

T. Liang (2022) first, and Fan et al. (2024) later, proposed a really interesting new model to compute prediction bands around a point predictor regression function. Using a representer theorem introduced by Marteau-Ferey et al. (2020), they estimate the variance function in a non-negative RKHS, using the residuals of the former mean regression function to assess the variance of the data. Regrettably, both their calibration procedure seems to be lacking. In the next chapter, we are going to investigate Marteau-Ferey et al. (2020) representer theorem and propose modifications on Liang's and Fan's models.

Chapter 5

Improving kernel methods

As we just uncovered, Liang's and Fan's models both present some drawbacks. In this chapter we shall present solutions to those problems and will propose a new model that has the same interesting properties of training-conditional coverage, flexibility, but with easy to tune hyperparameters, while being easy to compute.

5.1 Representation of non-negative functions

In this section we will present the representer theorem from Marteau-Ferey et al. (2020). We will focus on the primal formulation and investigate an equivalent finite-dimensional formulation in the primal. The goal of Marteau-Ferey et al. (2020) is to provide a non-parametric model for non-negative functions, within a framework that satisfies important proprieties that made the success of previous methods such as linear models.

5.1.1 A representer theorem

Again, we consider $\mathcal{H}$ our reproducible kernel Hilbert space and $\phi\colon \mathcal{X} \longrightarrow \mathcal{H}$ the feature map. Let $\mathcal{S}(\mathcal{H})$ bet the set of bounded Hermitian linear operator from $\mathcal{H}$ to $\mathcal{H}$. Our functions of interest is

$$v_{\mathcal{A}}(x) = \phi(x)^{\top}\mathcal{A}\phi(x), \text{ with } \mathcal{A} \in \mathcal{S}(\mathcal{H}),\ \mathcal{A} \succeq 0. \tag{5.1}$$

This function model satisfies the next proposition, which is necessary for our use case.

Proposition 1 (Non-negativity)**.** $\forall x \in \mathcal{X}$, $v_{\mathcal{A}}(x) \geq 0$.

We are interested in a problem which formulation is close to the one of linear models. Consider $L\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ a loss function and $\Omega\colon \mathcal{S}(\mathcal{H}) \to \mathbb{R}$ a regulariser function and define the following problem:

Problem 3.

$$\inf_{\mathcal{A}\succeq 0} L\left(v_{\mathcal{A}}(x_1), \ldots, v_{\mathcal{A}}(x_n)\right) + \Omega(\mathcal{A}). \tag{5.2}$$

This exact problem is studied extensively in Marteau-Ferey et al. (2020). It is shown that this framework has numerous qualities, also present in linear models. The main contribution of Marteau-Ferey et al. (2020) is the following theorem that gives a finite dimensional representation of the above Problem 3. It is this theorem that is used by T. Liang (2022) and Fan et al. (2024) in their papers to establish Theorems 6 and 11.

Remark 4 (Regulariser function)**.** *The regulariser function considered in Marteau-Ferey et al. (2020) is of the form*

$$\Omega(\mathcal{A}) = \lambda_1 \|\mathcal{A}\|_\star + \lambda_2 \|\mathcal{A}\|_F^2 \tag{5.3}$$

with $\lambda_1, \lambda_2 \geq 0$ *and* $\lambda_1 + \lambda_2 > 0$. Ω *is strongly convex when* $\lambda_2 > 0$.

Theorem 12 (A representer theorem for Problem 3)
Let $L\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ *a lower semi-continuous and bounded below loss function, and* $\Omega\colon \mathcal{S}(\mathcal{H}) \to \mathbb{R}$ *a regulariser function as in Remark 4. Problem 3 admits a solution* $A_\star$ *that can be written:*

$$A_\star = \sum_{i,j=1}^{n} B_{i,j} \phi(x_i)\phi(x_j)^\top, \text{ for some matrix } \boldsymbol{B} \in \mathbb{R}^{n\times n},\ \boldsymbol{B} \succeq 0. \tag{5.4}$$

Using the solution $A_\star$*, the associated non-negative function writes:*

$$v_{A_\star}(x) = \sum_{i,j=1}^{n} B_{i,j} k(x, x_i)k(x, x_j), \text{ for } \boldsymbol{B} \in \mathbb{R}^{n\times n},\ \boldsymbol{B} \succeq 0. \tag{5.5}$$

Moreover, if L *is convex and* $\lambda_2 > 0$*, then* $A_\star$ *is unique.*

Theorem 12 is extensively used in T. Liang (2022) and Fan et al. (2024). Unfortunately, as we saw in Sections 4.1.2 and 4.2.3, especially for Fan's model, the semi-definite programming solver has a hard time finding a solution. It is not clear why, but our practical experiment show that training the semi-definite problem is a time-consuming task. In the next section we explore an equivalent finite-dimensional representation of Theorem 12.

5.1.2 Equivalent formulation of the primal

Marteau-Ferey et al. (2020) proposes in their Proposition 3 an alternative equivalent finite dimensional formulation of Theorem 12. Let us dive into how to pass from Theorem 12 to the equivalent form. Our function from Equation (5.1) can be written, using the representer theorem, in the form of Equation (5.5). If we simply write for all $x \in \mathcal{X}$, $w(x) = (k(x, x_1), \ldots, k(x, x_n)))$, then $v_{A_\star}$ can be re-written:

$$v_{A_\star}(x) = w(x)^\top \boldsymbol{B} w(x) = v_{\boldsymbol{B}}(x). \tag{5.6}$$

Hence, $\boldsymbol{B}$ characterises the non negative function of interest v. Using this writing leads to Theorems 6 and 11 and this is why, in what follows, we call those "B formulations".

Now, $\boldsymbol{K}$ is the Gram matrix of the training data associated with a chosen kernel k. Let $\boldsymbol{V}$ be the upper factor of the Cholesky decomposition of $\boldsymbol{K}$, therefore $\boldsymbol{K} = \boldsymbol{V}^\top \boldsymbol{V}$. We are going to introduce $\boldsymbol{K}$ into Equation (5.6)[18]:

$$v_{\boldsymbol{B}}(x) = w(x)^\top \boldsymbol{V}^{-1} \boldsymbol{V}\boldsymbol{B}\boldsymbol{V}^\top \boldsymbol{V}^{-\top} w(x) \tag{5.7}$$

$$= \left[\boldsymbol{V}^{-\top} w(x)\right]^\top \boldsymbol{V}\boldsymbol{B}\boldsymbol{V}^\top \left[\boldsymbol{V}^{-\top} w(x)\right] \tag{5.8}$$

We can define the empirical feature map Φ:

$$\Phi(x) = \boldsymbol{V}^{-\top} w(x) \tag{5.9}$$

and a new positive definite matrix:

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{B}\boldsymbol{V}^\top. \tag{5.10}$$

[18] This is possible if and only if $\boldsymbol{V}$ is invertible, which is the case whenever it is full rank.

Henceforth, we can write:

$$v_{\boldsymbol{B}}(x) = \Phi(x)^\top \boldsymbol{A}\Phi(x) = \tilde{v}_{\boldsymbol{A}}(x) \tag{5.11}$$

The idea is to use this $\tilde{v}_{\boldsymbol{A}}$ function instead of $v_{\boldsymbol{B}}$ in all formulations. This is made possible thanks to the following proposition:

Proposition 2. *Under the assumptions of Theorem 12, the following problem admits a solution:*

$$\min_{\boldsymbol{A}\succeq 0} L\left(\tilde{v}_{\boldsymbol{A}}(x_1), \dots, \tilde{v}_{\boldsymbol{A}}(x_n)\right) + \Omega(\boldsymbol{A}), \quad \textit{where } \boldsymbol{A} \in \mathbb{R}^{n\times n},\ \boldsymbol{A} \succeq 0 \tag{5.12}$$

Moreover, for any given solution $\boldsymbol{A}_\star$ of Equation (5.12), $v_{\boldsymbol{A}_\star}$ is a solution of Problem 3.

In addition to the representer theorem, Marteau-Ferey et al. (2020) provide an equivalent finite-dimensional formulation. Our goal is to use this form instead of the "B formulation" to see if it can reduce the solver's optimisation time.

5.1.3 Using this equivalent form

Now let us use Proposition 2 to rewrite Theorem 6 and Theorem 11 using the "A formulation" from Equation (5.11).

Theorem 13 (Liang, formulation A)
Using Equation (5.11), Liang's model writes in finite form:

$$\min_{\boldsymbol{A}\in\mathbb{S}^n} \quad \mathrm{Tr}(\boldsymbol{A}) \tag{5.13a}$$

$$s.t.\ \ \tilde{v}_{\boldsymbol{A}}(x_i) \geq \left(y_i - \widehat{m}_p(x_i)\right)^2,\ i \in [\![1, n]\!] \tag{5.13b}$$

$$\boldsymbol{A} \succeq 0 \tag{5.13c}$$

Theorem 14 (Fan, formulation A)
Using Equation (5.11), Fan's model writes in finite form:

$$\min_{\boldsymbol{A}\in\mathbb{S}^n} \quad \mathrm{Tr}(\boldsymbol{K}\boldsymbol{A}) \tag{5.14a}$$

$$s.t.\ \ \tilde{v}_{\boldsymbol{A}}(x_i) \geq \left(y_i - \widehat{m}_p(x_i)\right)^2,\ i \in [\![1, n]\!] \tag{5.14b}$$

$$\mathrm{Tr}(\boldsymbol{A}) \leq r \tag{5.14c}$$

$$\boldsymbol{A} \succeq 0 \tag{5.14d}$$

This "A formulation" involves more steps to compute $\tilde{v}_{\boldsymbol{A}}(x)$ for any $x \in \mathcal{X}$ and in practice, one must always save both the matrices $\boldsymbol{A}$ and $\boldsymbol{V}$ in order to infer on a new data point. Figure 5.1 showcases the time needed for the semi-definite solver to find a solution for both Liang's and Fan's models and for both formulations. As we can see, formulation A (light blue) is much more usable than formulation B (orange). The solver converges both quicker and found an optimal solution more often.

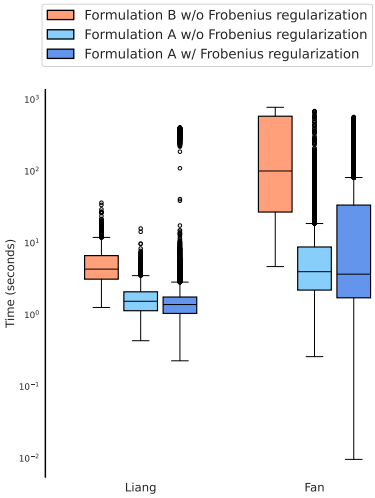


Figure 5.1: Time comparison between formulation A, B and Frobenius regularisation. Each model has been run for several training seeds and different hyperparameters setup. The y-axis is logarithmic scale.

5.1.4 On the regulariser function

We focus here on the regulariser function characteristics from Marteau-Ferey et al. (2020), diving deeper on Remark 4. The regulariser function they considered is:

$$\Omega(A) = \lambda_1 \|A\|_\star + \lambda_2 \|A\|_F^2 \ .$$

Both Liang and Fan only considered this regulariser function with $\lambda_2 = 0$. However, Marteau-Ferey sustains that $\lambda_2 > 0$ yields a strongly convex

[19] The loss function is indeed convex in both Liang (null function) and Fan (empirical mean).

regulariser function, and a unique solution to Problem 3, if the loss function L is convex[19]. Therefore, considering $\lambda_2 > 0$ in Liang's and Fan's models should, in theory, scale down the time needed to solve the SDP problem. The problems can be rewritten as:

Liang:

$$\min_{\boldsymbol{A}\in\mathbb{S}^n} \quad \mathrm{Tr}(\boldsymbol{A}) + \lambda_2 \mathrm{Tr}(\boldsymbol{A}^2) \tag{5.15a}$$

$$\text{s.t. } \tilde{v}_{\boldsymbol{A}}(x_i) \geq \left(y_i - \widehat{m}_p(x_i)\right)^2,\ i \in [\![1, n]\!] \tag{5.15b}$$

$$\boldsymbol{A} \succeq 0 \tag{5.15c}$$

Fan:

$$\min_{\boldsymbol{A}\in\mathbb{S}^n} \quad \mathrm{Tr}(\boldsymbol{K}\boldsymbol{A}) + \lambda_2 \mathrm{Tr}(\boldsymbol{A}^2) \tag{5.16a}$$

$$\text{s.t. } \tilde{v}_{\boldsymbol{A}}(x_i) \geq \left(y_i - \widehat{m}_p(x_i)\right)^2,\ i \in [\![1, n]\!] \tag{5.16b}$$

$$\mathrm{Tr}(\boldsymbol{A}) \leq r \tag{5.16c}$$

$$\boldsymbol{A} \succeq 0 \tag{5.16d}$$

We can see in Figure 5.1 that the gains are not as important as we expected. In order to understand why, we decided to investigate the impact of the regularisation for different values of λ_2. In total we considered 14 different values ranging from 10^{-10} to 10^6. As we can see in Figure 5.2, the benefit does not exist whatever the regularisation value. We see that for the smallest λ_2 values, the improvement exists but is incremental for both Fan and Liang. Then, for reasons that we cannot explain, the training time of Liang's problem goes up for values of 10^{-4} to 10^{-2} and then go down again, achieving a substantial improvement for values $1, 10, 100$. For higher values, the training time seems to tend toward the same training time as without the regularisation. In the case of Fan, the regularised model seems to train faster up to a regularisation value of 1. After that the training time explodes with the value of λ_2. An interesting fact about Fan's model is that the training time is heterogeneous across different p values. A bigger value for p seems to lead towards higher training

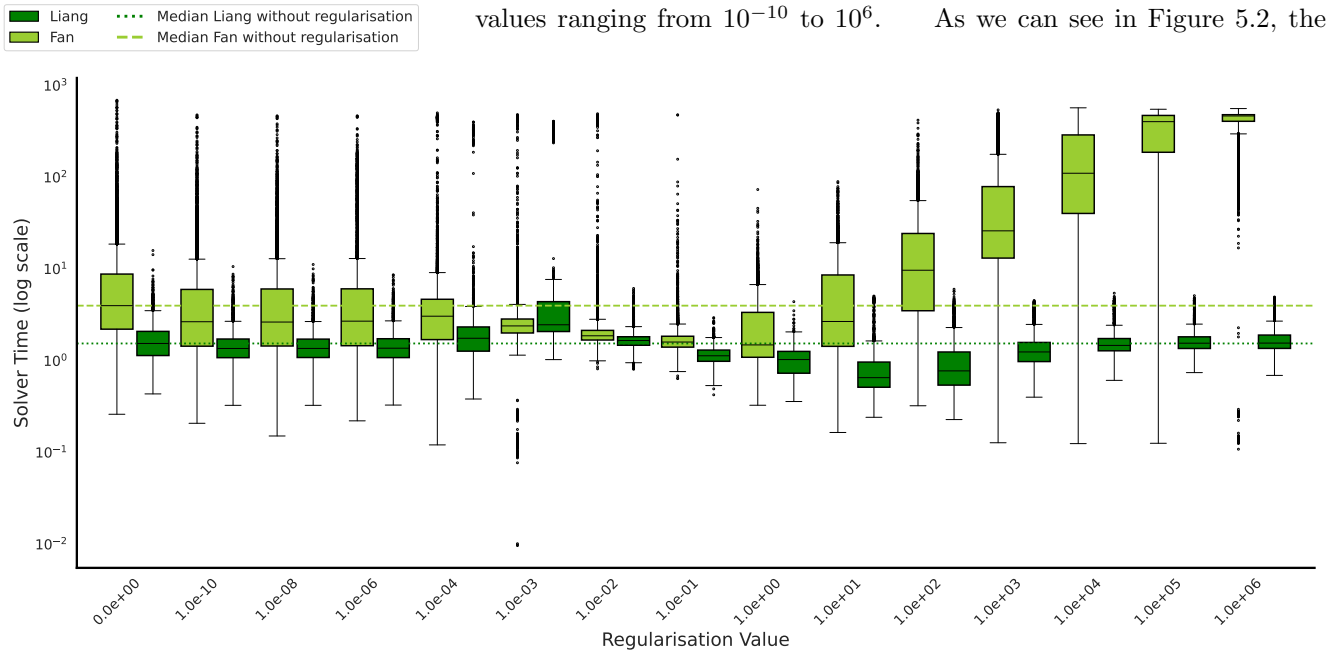


Figure 5.2: Time comparison between different regularisation values λ_2. Each model has been run for several training seeds and multiple length scales. The y-axis is logarithmic scale.

time. We leave this discussion as a research direction. As a general observation we can say that Liang's model is training faster than Fan's model.

This analysis shows that the impact of the Frobenius regularisation on the training time is not as straightforward as we initially thought. The choice of λ_2 cannot be underestimated. We did not come up with a sensible rule to choose its value, however, in our cases, a value of 1 seems to be beneficial in most cases. Therefore, for all the next experiments, we are going to use a value of $\lambda_2 = 1$. We believe that it is also important to keep in mind the fact that we only considered small problems. Indeed we only have here a hundred observations. In more complex settings, the Frobenius regularisation *might* stand out.

5.2 Refined calibration

In this section we are going to discuss the calibration procedure of the scale down parameter λ_α. We saw in Liang's model that the calibration procedure was sub-optimal, because the tuning of its hyperparameter is difficult and has a huge impact on the prediction bands shape. On Fan's side, the calibration procedure is also lacking. Considering a subset of the calibration dataset makes the bands too narrow and do not seem to guarantee, empirically, the coverage. Thus, we consider the following calibration procedure for the rest of the models and experiments. This

Algorithm 3: Calibration for Coverage Control

Input: Calibration set $\mathcal{D}_m = \{(x'_1, y'_1), \dots, (x'_n, y'_n)\}$, coverage level $1-\alpha$, $\widehat{m}_p$ and $\widehat{v}_n$ trained on respectively $\mathcal{D}_p$ and $\mathcal{D}_n$.

Output: Calibrated $\widehat{\lambda}_\alpha$;

Compute the $(1-\alpha)^{\text{th}}$ empirical quantile of $\frac{\left(y'_j - \widehat{m}_p(x'_j)\right)^2}{(\widehat{v}_n(x'_j)+\delta)}$:

$$\widehat{\lambda}_\alpha \coloneqq \inf_{\lambda}\left\{\lambda \colon \frac{1}{m}\sum_{j=1}^{m} \mathbb{1}_{\left[\left(y'_j - \widehat{m}_p(x'_j)\right)^2 > \lambda\left(\widehat{v}_n(x'_j)+\delta\right)\right]} \leq \alpha\right\} \tag{5.17}$$

return $\widehat{\lambda}_\alpha$.

algorithm is the same as Fan's, but *without* the subsetting. It is simple and enables to widen the prediction bands or narrow them as needed. Furthermore, this algorithm supports our theoretical claims that the model we propose have train-conditional coverage guarantee. Using formulation A and the calibration Algorithm 3, we construct the prediction bands as follow:

$$\widehat{C}_{(p,n,m)}(x,\delta) = \left[\widehat{m}_p(x) \pm \sqrt{\widehat{\lambda}_{m,\alpha} \cdot \left(\widehat{\widehat{v}}_n(x) + \delta\right)}\right]. \tag{5.18}$$

5.3 Illustrative comparison

We are now ready to test our full model, that is solving Equations (5.15a) and (5.16a) and calibrating the prediction bands using Algorithm 3.

Experiment 2. With the new calibration procedure, "Liang's formulation A" problem only has the length scale of the kernel and the Frobenius norm coefficient as hyperparameters. We stick to the exact same Gaussian

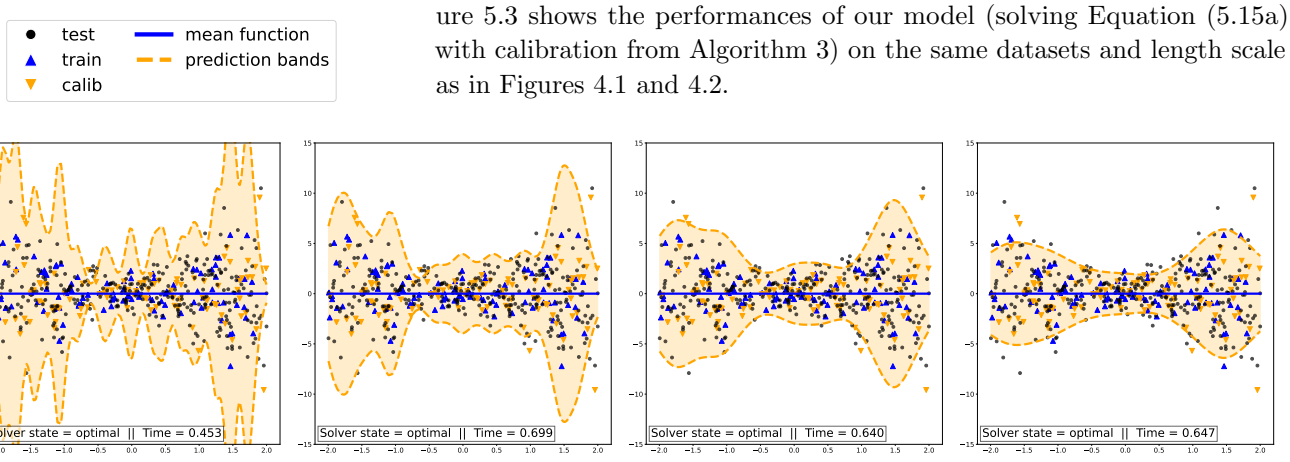
Fig	$\widehat{\lambda}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 5.3a	7.6	1.0	0.95	0.95	15.4
Figure 5.3b	2.6	1.0	0.95	0.94	10.9
Figure 5.3c	1.4	1.0	0.95	0.92	9.4
Figure 5.3d	0.69	0.91	0.95	0.84	7.4

Table 5.1: Statistics about experiments in Figure 5.3.

Fig	p	$\widehat{\lambda}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 5.4a	2	9.2	1.0	0.95	0.94	16.5
Figure 5.4b	2	3.4	1.0	0.95	0.96	12.9
Figure 5.4c	2	1.6	1.0	0.95	0.92	10.0
Figure 5.4d	2	1.2	1.0	0.95	0.93	9.5
Figure 5.4e	100	7.5	1.0	0.95	0.94	17.6
Figure 5.4f	100	5.9	1.0	0.95	0.95	17.9
Figure 5.4g	100	1.6	1.0	0.95	0.93	10.1
Figure 5.4h	100	1.2	1.0	0.95	0.93	9.5

Table 5.2: Statistics about experiments in Figure 5.4.

process previously trained as the estimation of the mean function. Figure 5.3 shows the performances of our model (solving Equation (5.15a) with calibration from Algorithm 3) on the same datasets and length scale as in Figures 4.1 and 4.2.

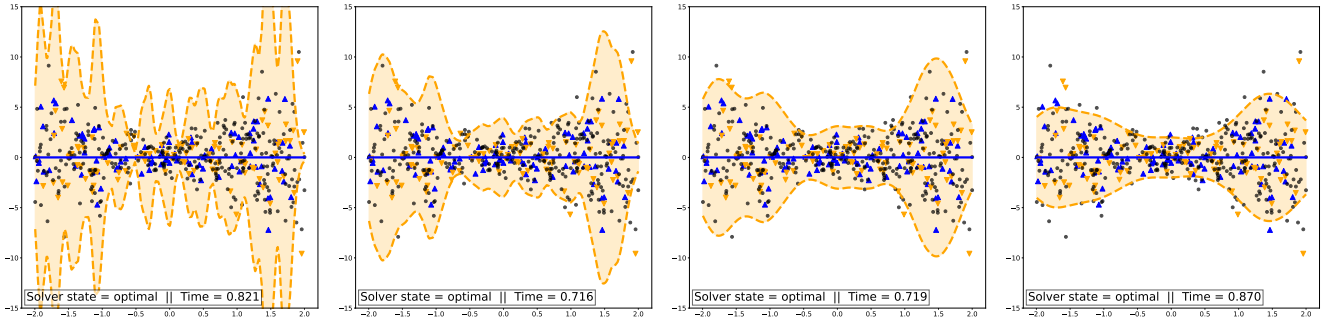


(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

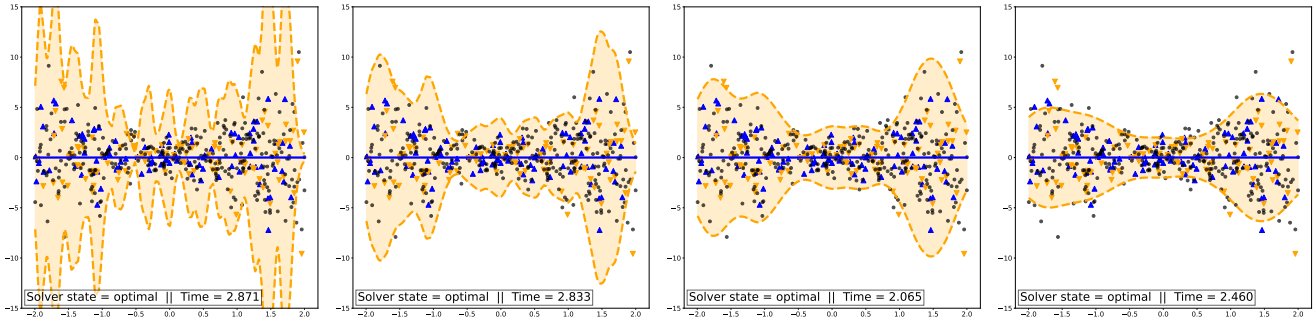
Figure 5.3: Experiment 2 - Liang A - $\lambda_2 = 1$; $\delta = 0.001$ - Evolution of the prediction bands shape with regard to the length scale.

The results are much better than formulation B, both in terms of coverage and in term of hyperparameter tuning. We can now compare Fan's model on a more fair basis as the single hyperparameter that changes from Liang to Fan is r. Again to choose it we used the result of Liang's problem and applied a coefficient p. Figure 5.4 shows Fan's model for $p = 2$ in the top row and $p = 100$ in the bottom row. The differences between $p = 2$ and $p = 100$ are slight. Only for small enought length scale do we notice a difference, the difference being the larger r is the more the prediction bands pinches.
Between Liang's and Fan's formulation, Fan's prediction bands seems to be overall a little more wide (bigger mean width). One possible explanation is that Fan's prediction bands pinches more than Liang's and thus during calibration, the quantile gets bigger in order to get the necessary calibration points within the bands, especially if a number of calibration points are near a pinch point. Table 5.2 supports this claim.

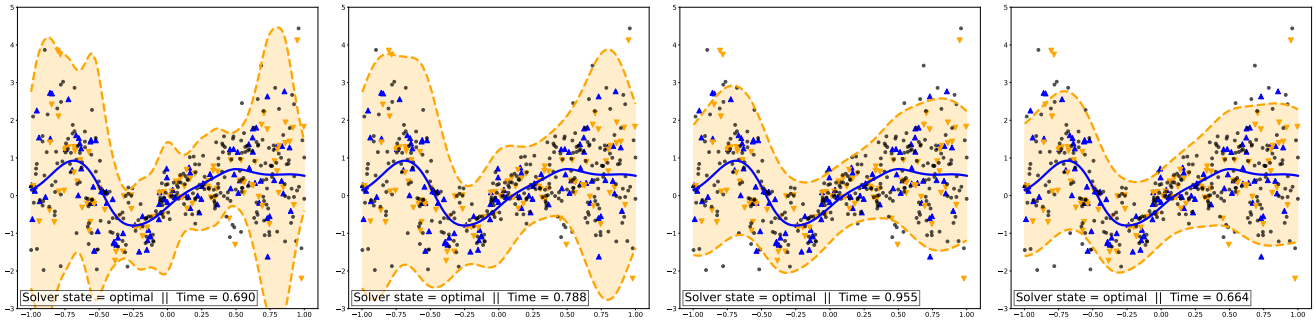
Experiment 4. Let us explore how our models perform on experiment four, without a null mean function. Once again, we consider the Gaussian process studied before as our mean regressor function. Liang's formulation



(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6



(e) length scale = 0.1 **(f)** length scale = 0.2 **(g)** length scale = 0.4 **(h)** length scale = 0.6



(a) length scale = 0.1 **(b)** length scale = 0.2 **(c)** length scale = 0.4 **(d)** length scale = 0.6

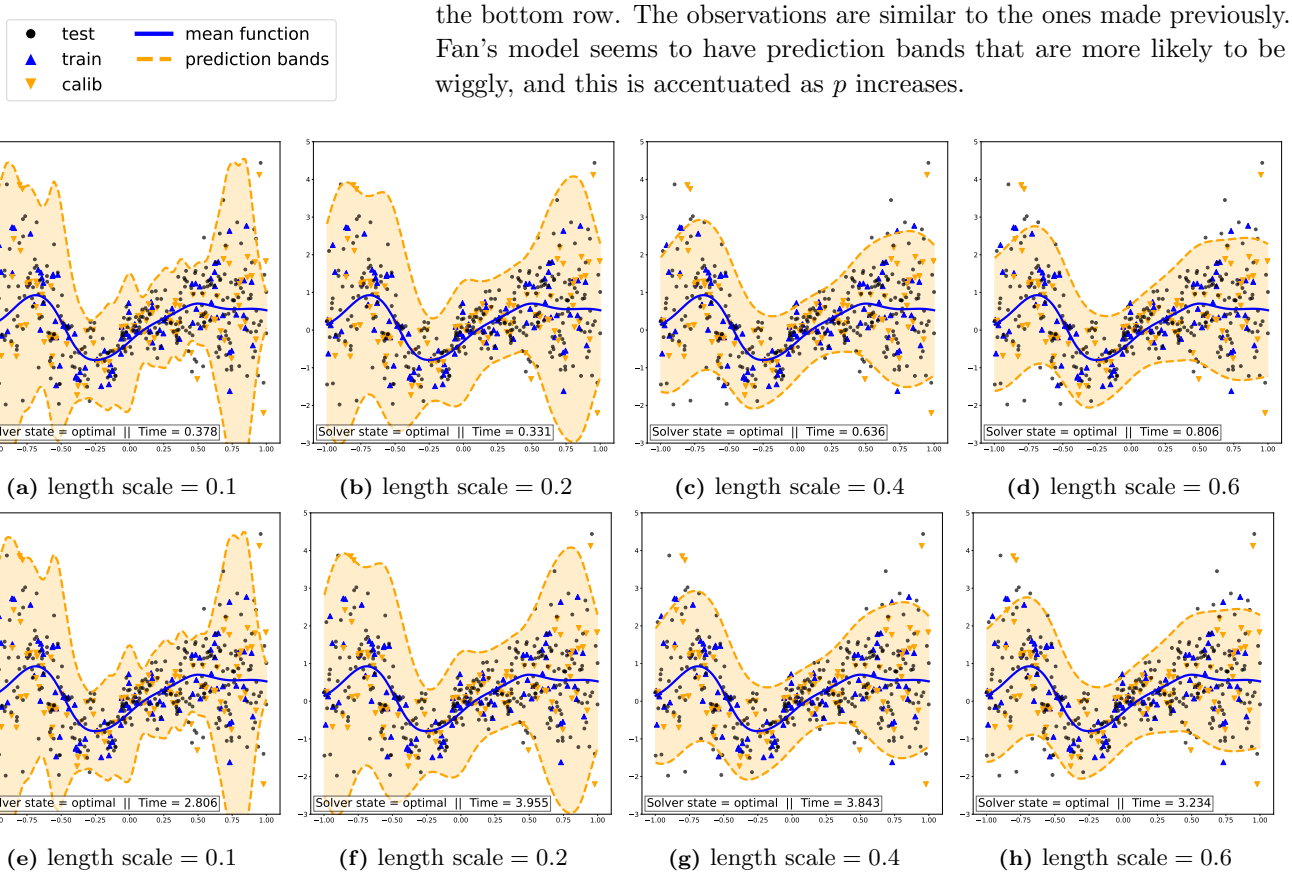
Figure 5.4: Experiment 2 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$ - Evolution of the prediction bands shape with regard to the length scale. Top row is $p = 2$ and bottom row is $p = 100$.

Figure 5.5: Experiment 4 - Liang A - $\lambda_2 = 1$; $\delta = 0.001$ - Evolution of the prediction bands with regard to the length scale.

Fig	$\widehat{\lambda}_\alpha$	Train Cov.	Cal. Cov.	Test Cov.	Avg. length
Figure 5.5a	7.6	1.0	0.95	0.95	15.4
Figure 5.5b	2.6	1.0	0.95	0.94	10.9
Figure 5.5c	1.4	1.0	0.95	0.92	9.4
Figure 5.5d	0.69	0.91	0.95	0.84	7.4

Table 5.3: Statistics about experiments in Figure 5.5.

seems to perform really well, across all length scales. With the calibration procedure of Algorithm 3 we have one less hyperparameter to tune. We will see in Remarks 5 and 6 that λ_2 and δ are not that important, at least for the shape of the bands. Therefore, Liang's model leaves a single hyperparameter to tune, the length scale of the variance function. Multiple length scales are presented Figure 5.5. Its tuning is the objective of the next section. Before that, let us explore the performances of Fan's formulation on this experiment four. Once again, we set r as a coefficient of the result of Liang's. We will present, in Figure 5.6, $p = 2$ in the top row and $p = 100$ in the bottom row. The observations are similar to the ones made previously. Fan's model seems to have prediction bands that are more likely to be wiggly, and this is accentuated as p increases.



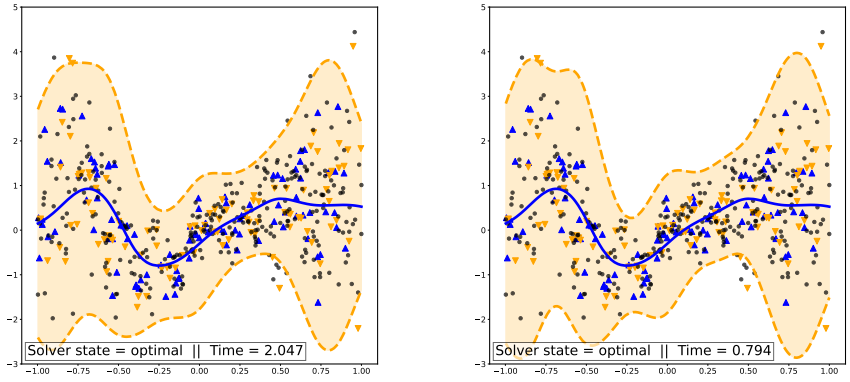
(a) length scale $= 0.1$ **(b)** length scale $= 0.2$ **(c)** length scale $= 0.4$ **(d)** length scale $= 0.6$

(e) length scale $= 0.1$ **(f)** length scale $= 0.2$ **(g)** length scale $= 0.4$ **(h)** length scale $= 0.6$

Figure 5.6: Experiment 4 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$ - Evolution of the prediction bands shape with regard to the length scale. Top row is $p = 2$ and bottom row is $p = 100$.

Remark 5 (On the impact of λ_2 on the prediction bands.)**.** *This hyperparameter is new from the original proposition of Liang and Fan. In Figure 5.1 we already compared the impact of having λ_2 strictly positive (darker blue) and $\lambda_2 = 0$ (lighter blue). Here what we want to observe is the impact on the shapes of the predictions bands. We consider Liang's problem Equation* (5.16a) *on experiment four with a length scale of* 0.2*. Figure 5.7 shows the differences between $\lambda_2 = 1e^{-3}$ and $\lambda_2 = 100$. In the shapes, we can see slight differences (top left of the figures) but nothing striking.*

Remark 6 (On the impact of δ on the prediction bands.)**.** *The δ parameter appears only at the calibration stage. Its impact on the prediction bands is exactly the same as in Fan's original model. Thus the conclusion we make here are very similar to Remark 3. To illustrate this impact we are going to consider Fan's model with $p = 100$ and a length scale of* 0.6*. Figure 5.8 compares four values, $\delta = 1e^{-8}$, $\delta = 1$, $\delta = 10$ and $\delta = 100$. We observe that as δ grows the bands become of constant width. One important observation is that between $\delta = 1e^{-8}$ in Figure 5.8a and $\delta = 1e^{-3}$ in Figure 5.5b, the differences are almost absent. Therefore if we choose δ*



(a) $\lambda_2 = 1$ **(b)** $\lambda_2 = 100$

Figure 5.7: Experiment 4 - Liang A - $ls = 0.2$; $\delta = 0.001$ - Evolution of the prediction bands shape with regard to the regularisation value λ_2.

small enough, we will not impact the prediction bands.

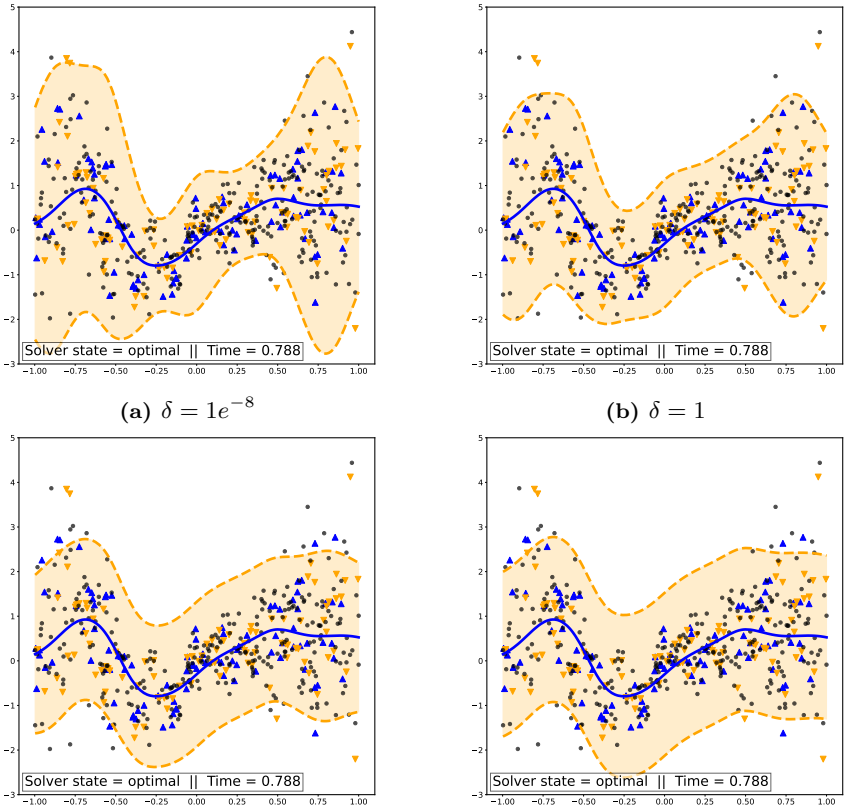
5.4 Hyperparameter tuning

In Liang's formulation A there is a single hyperparameter to tune, the length scale of the variance kernel function. In Fan's version, we also have to choose a value for r. We did not come up with a better solution than to estimate r with the solution of Liang's model. Therefore it appears that Fan's version is harder to tune than Liang's. We are going to set aside the tuning of r and focus ourselves on the length scale of the variance kernel. As we saw in Figures 5.5 and 5.6, this length scale has a huge impact on the shape of the prediction bands. Let us discuss a way to choose the best length scale, without knowing the shape of the prediction bands.

Our goal is to find the prediction bands that are the smallest (in term of average width) while having the right coverage, and while being adaptive to the data. First of all, we do not need to pay attention to the coverage guarantee because the theorem supporting the model claims the following: whatever the good or bad specification of our model, for a large enough training sample, the training conditional coverage is met. In our case this implies that the length scale does not impact the coverage guarantee. Therefore, the choice of the length scale is going to affect only the average width of the bands and their adaptivity. As we can see in Figure 5.3 the average width of the prediction bands (on the calibration dataset) can be very low for prediction bands that are flat. Thus it is not a good measure to use to choose the length scale.

What we are looking for is prediction bands that are *adaptive* to the data. We want the bands to be narrow as the uncertainty decreases and wide when the uncertainty is high. The uncertainty in the predictions can be measured through the residuals. Therefore we want to have large bands when the residual is high and narrow bands when the residual is low. In other word, we seek a model that has a high dependence between the residuals and the width of the prediction bands. Using the right metric, we will choose the length scale that produces the predictions bands the most dependent to the residuals.

A key ingredient here is the metric we are going to use to measure this



(a) $\delta = 1e^{-8}$ **(b)** $\delta = 1$

(c) $\delta = 10$ **(d)** $\delta = 100$

Figure 5.8: Experiment 4 - Liang A - $ls = 0.2$; $\lambda_2 = 1$ - Evolution of the prediction bands shape with regard to δ.

dependency. We now review a few of them, discussing their advantages and drawbacks. We study those metrics on ten length scales, from 0.1 to 1.0, on the following three models:

- Liang: solve Equation (5.15a) and calibrate using Algorithm 3, with $\lambda_2 = 1$ and $\delta = 1e^{-3}$.
- Fan with $p = 2$: solve Equation (5.16a) and calibrate using Algorithm 3, with $\lambda_2 = 1$ and $\delta = 1e^{-3}$.
- Fan with $p = 100$: solve Equation (5.16a) and calibrate using Algorithm 3, with $\lambda_2 = 1$ and $\delta = 1e^{-3}$.

First, let us look at the Pearson coefficient. This is the first coefficient one might think of to measure correlation.

Definition 6 (Pearson)**.** *Let (X, Y) be two random variables. The Pearson correlation coefficient is defined as:*

$$\rho_p(X, Y) = \frac{Cov(X, Y)}{\sqrt{Var(X)\, Var(Y)}} \tag{5.19}$$

Figure 5.9 shows the evolution of the Pearson coefficient, between the residuals and the prediction bands width, with regard to the length scale. The red vertical bars are for the highest values. As we can see, the coefficient seems to plateau and it is hard to clearly discriminate length scales. Indeed this measures only linear correlation, and what we are looking for is not necessarily linear. To adjust for that, one might look at the Spearman's rank correlation coefficient.

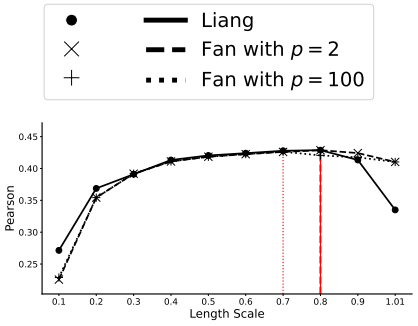


Figure 5.9: Pearson coefficient between residuals and prediction bands width, on calibration data.

Definition 7 (Spearman)**.** *Let (X, Y) be two random variables. We denote by $R(X)$ the ranks of the random variables. The Spearman correlation coefficient is defined as:*

$$\rho_s(X, Y) = \frac{Cov(R(X), R(Y))}{\sqrt{Var(R(X))\, Var(R(Y))}} \tag{5.20}$$

The same observations can be made about the Spearman coefficient. In Figure 5.10 we can see that the Spearman coefficient chooses the length scale 0.2. But it is hard to discriminate length scales based on that metric. Length scales 0.3 to 0.9 have really similar Spearman coefficient value. Finally, the Spearman coefficient, with regard to the length scale is not concave, which makes the length scale choice harder. We thus turned ourselves to a different kind of measure.

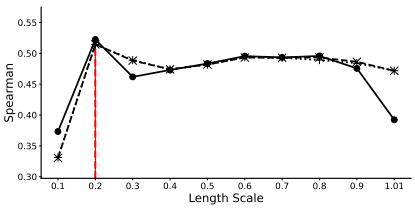


Figure 5.10: Spearman coefficient between residuals and prediction bands width, on calibration data.

A very common measure to evaluate dependency is the mutual information. It takes its root in information theory and has a very profound interpretation in simple cases, as the quantity of information obtained about one random variable by observing another one. A rigorous definition is as follow.

Definition 8. *Let (X, Y) be two random variables. We denote by P_X and P_Y the marginal distribution of those random variables, and $P_{(X,Y)}$ their joint distribution. The mutual information is defined as:*

$$MI(X, Y) = D_{KL}\left(P_{(X,Y)} \| P_X P_Y\right) \tag{5.21}$$

where D_{KL} is the Kullback-Leibler divergence.

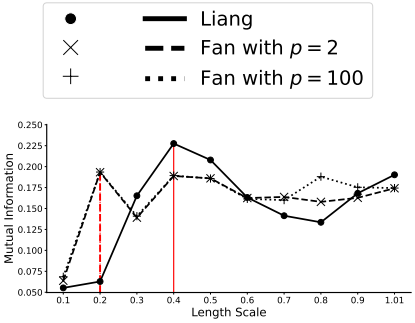


Figure 5.11: Mutual Information between residuals and prediction bands width, on calibration data.

Thus the mutual information (MI) is a measure of similarity between the joint distribution and the product of the marginal distributions[20]. If the variables are not correlated at all, then the product of the marginal and the joint distribution are very different and the mutual information is null. If the variable are correlated then the mutual information must be high. We refer to Kraskov et al. (2004) regarding algorithms to compute the mutual information. This method uses neighbours, and one hyperparameter is the number of neighbours to use in the computation. We did not investigate its impact here and chose 5 for all experiments. Figure 5.11 shows the evolution of the mutual information between the residuals and the width of the bands (on the calibration dataset) against the length scale. This metric is much more discriminating than the Spearman and Pearson coefficients. Although it works well for Liang's model, selecting a length scale of 0.4 (see Figure 5.5c), the selection fails for Fan's model. Moreover, the evolution of the mutual information does not allow for a distinct choice of length scale. What makes the mutual information work for Liang but not for Fan even thought the prediction bands have similar shapes? Is the computation of mutual information, in our case, sensible to very slight variations? We leave this study as a future research direction.

[20] There exists a lot of other metrics that compares the joint distribution to the product of the marginals, like f-Divergence, Wasserstein Dependency Measure and HSIC to name a few.

Finally, we tried one last metric to compute the dependence of the width and the residuals. This metric is the Hilbert-Schmidt Independence Criterion (HSIC), introduced by Gretton et al. (2005). The HSIC is a kernel based measure of dependence between two random variables. It leverages the RKHS theory to quantify how dependent those variables are, and can therefore identify non-linear relationships. We introduce two kernels k and l associated to two RKHS spaces $\mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$. Now consider the joint distribution of two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, P_{XY}.

The cross-covariance operator associated to this distribution is the linear operator between $\mathcal{H}_\mathcal{X}$ and $\mathcal{H}_\mathcal{Y}$ defined for every $f \in \mathcal{H}_\mathcal{X}$ and $g \in \mathcal{H}_\mathcal{Y}$:

$$\langle f, C_{XY} g \rangle_{\mathcal{H}_\mathcal{X}} = \mathbb{E}_{XY}[f(X)g(Y)] - \mathbb{E}_X f(X)\mathbb{E}_Y g(Y). \tag{5.22}$$

The HSIC between X and Y, considering the two above RKHS, is the Hilbert-Schmidt norm of this cross-covariance operator:

$$HSIC(X,Y)_{\mathcal{H}_\mathcal{X},\mathcal{H}_\mathcal{Y}} = ||C_{XY}||^2_{HS}. \tag{5.23}$$

Gretton et al. (2005) showed that Equation (5.23) can be written in terms of kernels. A very nice feature of the HSIC is that it can also be written using the Maximum Mean discrepancy (MMD) distance. This distance has been introduced in distribution comparison by Gretton et al. (2012).

$$HSIC(X,Y)_{\mathcal{H}_\mathcal{X},\mathcal{H}_\mathcal{Y}} = MMD^2\left(P_{XY}, P_X P_Y\right) \tag{5.24}$$
$$= ||m^e_{XY} - m^e_X \otimes m^e_Y||^2_{\mathcal{H}_\mathcal{X}\otimes\mathcal{H}_\mathcal{Y}} \tag{5.25}$$

where $m^e_{(\cdot)}$ are the mean embedding of the associated variable. This formulation of the HSIC bonds it with the mutual information as a distance metric between the joint probability and the product of the marginal probabilities. However, the HSIC comes with one significant drawback, this is the choice of the kernels, and more specifically the choice of their hyperparameters. Indeed, it seems senseless to use a metric to choose the length scale of a kernel, that needs itself two different length scales. That is why we decided to use the Energy kernel for both variables.

Definition 9 (Energy kernel)**.** *Let $\mathcal{X} \subset \mathcal{H}$, $\mathcal{H}$ a Hilbert space. The Energy kernel $k_E \colon \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is given by:*

$$k_E(x, x') = ||x|| + ||x'|| - ||x - x'||.$$

This makes it a very easy to use kernel. Furthermore, it verifies a property needed for using a kernel with the MMD, which is the universal property. Figure 5.12 shows the performance of this metric on ten length scales. First of all, the length scales chosen for all models (Liang, Fan with $p = 2$ and Fan with $p = 100$) are all correct visually. Second, we observe that the metric seems to be strictly decreasing as the length scale increases. This is highly desirable in our case. The mutual information, even if it selected a correct length scale for Liang's model, has a spike at length scale 0.9 that is inexplicable. The HSIC with energy kernel seems to perform better on our toy experiment. More investigation needs to be persued in order to determine the robustness of this metric across multiple variations, but we will leave this as a research direction.

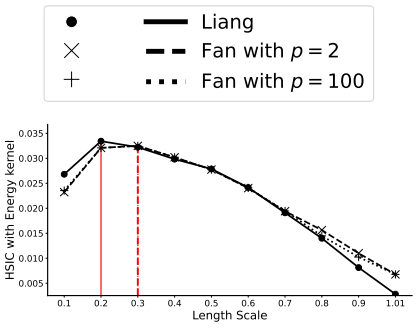


Figure 5.12: Energy-HSIC between residuals and prediction bands width, on calibration data.

Summing up. What we presented in this chapter is first an equivalent formulation of the problems using a result from Marteau-Ferey et al. (2020), which showed to be much more efficient in training-time. Second, we proposed to use a regularisation to make the semi-definite problem strongly convex to further enhance the feasibility, however it did not improve the training-times as expected. Third, we proposed a different calibration procedure from both Liang's and Fan's propositions. Ours, empirically, performs better and is more consistent, while keeping the training-conditional coverage guarantee. Fourth, we brought forward a metric, the Energy HSIC, to select the length scale of the variance kernel, with good results.

Our model keep all the very good properties of conformal prediction: distribution free, agnostic to the machine learning model and train-conditional guarantee. In the next chapter, we present yet another advantage of our model against conformal prediction, which is estimating simultaneously the conditional mean and the conditional variance.

Chapter 6

Kernel methods to estimate the mean and variance at once

Since now, we have only studied what we can call a "sequential" model. Indeed the first step was always to fit a point regressor on a first dataset (we used Gaussian processes, but any model can be used), and then fit the prediction bands around this point regressor to construct the set regressor. However, both T. Liang (2022) and Fan et al. (2024) write about a kernel model that can estimate the mean point predictor and the regression bands simultaneously. In the interest of keeping this report short, we are going to develop this model with the "A formulation" from Section 5.1, without going in depth on the propositions of T. Liang (2022) and Fan et al. (2024).

6.1 Conjectures for a representer theorem

First, let us introduce some notations. We will use superscripts m and v to denote the feature map, kernel and RKHS used for the estimation of the mean and the prediction bands respectively.

Problem 4 (Liang, simultaneous estimation)**.**

$$\min_{m\in\mathcal{H}^m,\mathcal{A}\in\mathcal{S}(\mathcal{H}^v)} \quad ||\mathcal{A}||_* + \lambda_2||\mathcal{A}||_F^2 \tag{6.1a}$$

$$s.t. \quad \langle\phi^v_{x_i}, \mathcal{A}(\phi^v_{x_i})\rangle_{\mathcal{H}^v} \geq \left(y_i - \langle\phi^m_{x_i}, m\rangle_{\mathcal{H}^m}\right)^2,\ i \in [\![1,n]\!] \tag{6.1b}$$

$$||m||_{\mathcal{H}^m} \leq s \tag{6.1c}$$

$$\mathcal{A} \succeq 0 \tag{6.1d}$$

Problem 5 (Fan, simultaneous estimation)**.**

$$\min_{m\in\mathcal{H}^m,\mathcal{A}\in\mathcal{S}(\mathcal{H}^v)} \quad \frac{1}{n}\sum_{i=1}^{n}\langle\phi^v_{x_i}, \mathcal{A}\phi^v_{x_i}\rangle_{\mathcal{H}^v} + \lambda_2||\mathcal{A}||_F^2 \tag{6.2a}$$

$$s.t. \quad \langle\phi^v_{x_i}, \mathcal{A}(\phi^v_{x_i})\rangle_{\mathcal{H}^v} \geq \left(y_i - \langle\phi^m_{x_i}, m\rangle_{\mathcal{H}^m}\right)^2,\ i \in [\![1,n]\!] \tag{6.2b}$$

$$||\mathcal{A}||_* \leq r \tag{6.2c}$$

$$||m||_{\mathcal{H}^m} \leq s \tag{6.2d}$$

$$\mathcal{A} \succeq 0 \tag{6.2e}$$

Now, those two problems are infinite dimensional: is there a way to make them tractable? The answer lies in the following conjectures, which are extensively used in both papers, but not proven. In the conjectures,

we present the representer theorem using the equivalent formulation of the primal, presented in Section 5.1.2.

Conjecture 1 (Representer theorem for Liang simultaneous estimation)

$$\min_{\boldsymbol{\gamma}\in\mathbb{R}^n, \boldsymbol{A}\in\mathbb{S}^n} \quad \operatorname{Tr}(\boldsymbol{A}) + \lambda_2 \operatorname{Tr}(\boldsymbol{A}^2) \tag{6.3a}$$

$$s.t. \;\; \tilde{v}_{\boldsymbol{A}}(x_i) \geq \left(y_i - \langle \boldsymbol{K}^m_{x_i}, \boldsymbol{\gamma}\rangle\right)^2, \; i \in [\![1, n]\!] \tag{6.3b}$$

$$\boldsymbol{\gamma}^\top \boldsymbol{K}^m \boldsymbol{\gamma} \leq s \tag{6.3c}$$

$$\boldsymbol{A} \succeq 0 \tag{6.3d}$$

Conjecture 2 (Representer theorem for Fan simultaneous estimation)

$$\min_{\boldsymbol{\gamma}\in\mathbb{R}^n, \boldsymbol{A}\in\mathbb{S}^n} \quad \operatorname{Tr}(\boldsymbol{K}^v \boldsymbol{A}) + \lambda_2 \operatorname{Tr}(\boldsymbol{A}^2) \tag{6.4a}$$

$$s.t. \;\; \tilde{v}_{\boldsymbol{A}}(x_i) \geq \left(y_i - \langle \boldsymbol{K}^m_{x_i}, \boldsymbol{\gamma}\rangle\right)^2, \; i \in [\![1, n]\!] \tag{6.4b}$$

$$\operatorname{Tr}(\boldsymbol{A}) \leq r \tag{6.4c}$$

$$\boldsymbol{\gamma}^\top \boldsymbol{K}^m \boldsymbol{\gamma} \leq s \tag{6.4d}$$

$$\boldsymbol{A} \succeq 0 \tag{6.4e}$$

We will not prove such conjectures here, but we will study the empirical behaviour of this model. The prediction bands are constructed using the calibration presented in Section 5.2 and writes for $x \in \mathcal{X}$:

$$\widehat{C}_{(n,m)}(x, \delta) = \left[\widehat{m}_n(x) \pm \sqrt{\widehat{\lambda}_{m,\alpha}\left(\widehat{v}_{n(x)} + \delta\right)}\right]. \tag{6.5}$$

It introduces new hyperparameters, the length scale of kernel $\boldsymbol{K}^m$ and the constraint parameter on the norm of the mean function, s. The length scale of the prediction bands will be chosen using the Energy-HSIC as presented in Section 5.4. The length scale for the mean function estimation will be chosen via a proxy that we detail later. For the constraint s, its choice is more subtle[21]. In the next section, we will illustrate how those two new models behave and we will expand on a procedure to find proxy values for those hyperparameters.

[21] A grid search would be unrealistic for two reasons. First, the computation time would explode. Second, we do not have any information on the range of values to try.

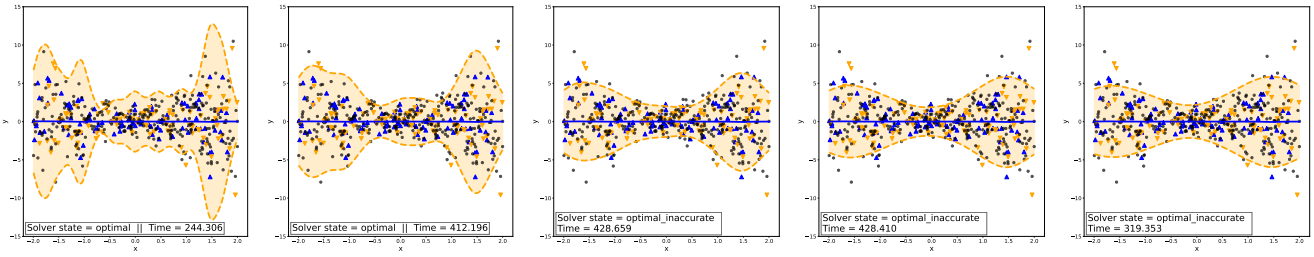
6.2 Empirical study

Before testing our new model against our toy datasets, let us consider a procedure to determine values for the new hyperparameters s and the length scale for the mean function. One possibility is to use a Gaussian process (on a separate dataset) to approximate those values. First of all, for the length scale, we simply use the posterior length scale of the Gaussian process. For the hyperparameter s, we retrieve the posterior coefficients $\boldsymbol{\gamma}_{\mathrm{GP}}$ and then define s as $\boldsymbol{\gamma}_{\mathrm{GP}}^\top \boldsymbol{K}^m \boldsymbol{\gamma}_{\mathrm{GP}}$, where $\boldsymbol{K}^m$ is the kernel matrix with the training data for solving the semi-definite problem.

Let us discuss a few hyperparameters that we are going to fix in this empirical study. Concerning the calibration procedure of those models, we will keep the one presented in Section 5.2. This adds a hyperparameter, δ and we will fix it to $\boldsymbol{\delta = 0.001}$. As discussed in Section 5.1.4, the impact of the λ_2 value in the shape of the prediction bands is once again hardly visible. Therefore we will not showcase its impact here, and will default to $\boldsymbol{\lambda_2 = 1}$. Finally, for Fan's formulation, we have one more hyperparameter, p. For ease of read, and because we studied it earlier, we are going to fix its value to $\boldsymbol{p = 2}$.

Experiment 2.

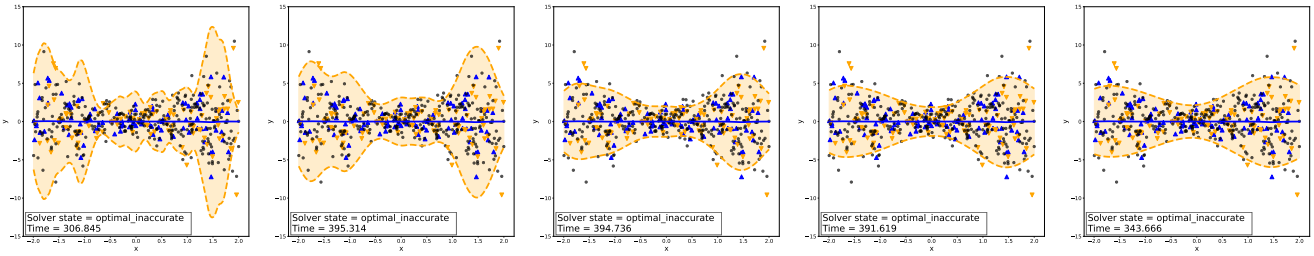
Liang. For Liang's model from Equation (6.3a), the results are similar to the "sequential" model, where the semi-definite problem was used to estimate the prediction bands only. The results are presented for a selection of length scale values in Figure 6.1. Indeed, the simultaneous model seems to estimate decently the null mean function, regardless of the value for the prediction bands length scale. Concerning the prediction bands themselves, their shape, coverage and average length do no change much compared to the sequential procedure.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure 6.1: Experiment 2 - Liang A, simultaneous - $\lambda_2 = 1; \delta = 0.001$ - Evolution of the mean function and prediction bands shape with regard to the length scale.

Fan. Taking a look at Fan's model from Equation (6.4a), the results are again similar to the "sequential" model. The null mean function is correctly estimated and thus the prediction bands are correct too. Note however that for all length scales, the SDP solver did not converge perfectly. The solutions are always "optimal inaccurate".



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure 6.2: Experiment 2 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$;$p = 2$ - Evolution of the mean function and the prediction bands shape with regard to the length scale.

Experiment 4.

Liang. Let us first look at Liang's model from Equation (6.3a). As we can see in Figure 6.3, changing the length scale for the prediction bands kernel has an impact not only on the prediction bands, but also on the estimation of the mean function[22]. This is really a side effect of estimating both functions simultaneously. We refer to Appendix E for a solution to this behaviour. In Figure 6.4 we showcase the evolution of the Energy-HSIC for ten length scale values. We can see that the metric behave as expected. The curve is nice and concave, allowing us to select a length scale without a doubt. The length scale selected for Liang's model is 0.4, see Remark 7.

Fan. Fan's model from Equation (6.4a) also has difficulties estimating the mean function. Despite Figure 6.5b estimating the prediction bands

[22] Here the length scale for the mean function is constant, indeed we used the same Gaussian process to estimate the hyperparameters.

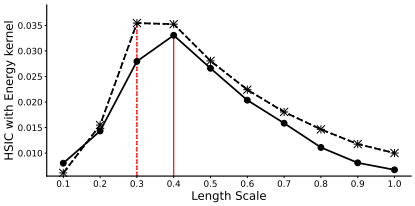
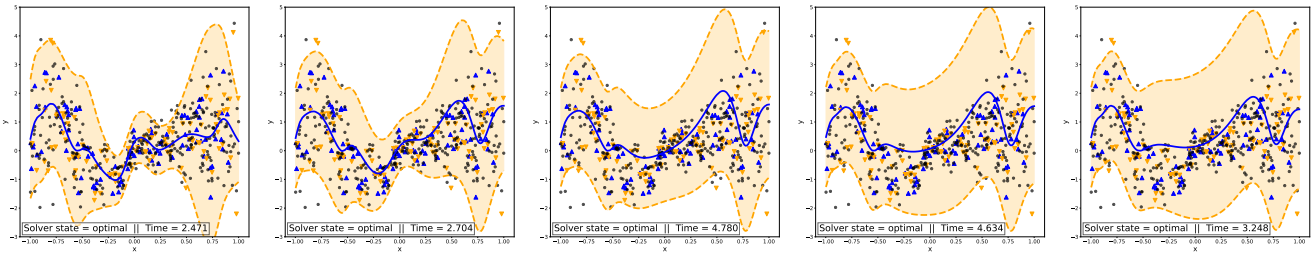
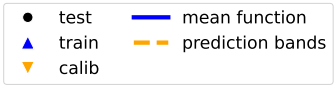


Figure 6.4: Energy-HSIC between residuals and prediction bandss width, on calibration data in Figure 6.3.

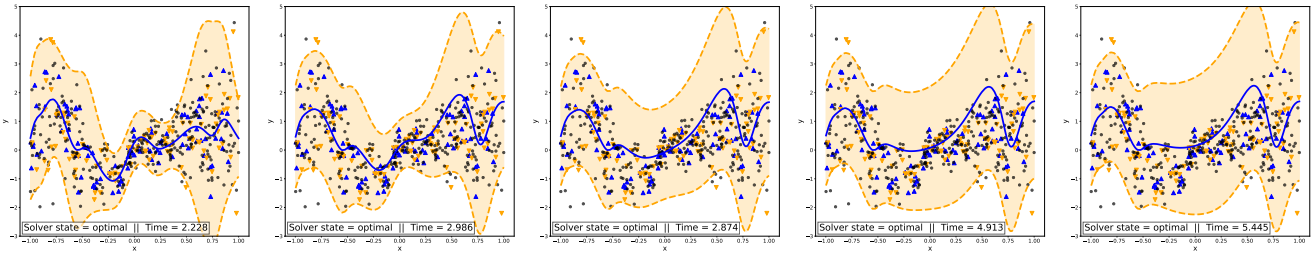
Figure 6.3: Experiment 4 - Liang A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$ - Evolution of the mean function and prediction bands shape with regard to the length scale.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0



well (visually only), we face the same side effect as in Liang's model, as the length scale increases the estimation of the mean function worsen. The Energy-HSIC still performs as expected, see Figure 6.4 and Remark 7.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure 6.5: Experiment 4 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2$ - Evolution of the mean function and the prediction bands shape with regard to the length scale.

Remark 7. *In Figure 6.3 we can see that to the naked eye, one would choose a length scale of* 0.2 *rather than the* 0.4 *selected by the Energy-HSIC. Yet, if we choose a finer length scale scale we can see that the Energy-HSIC indeed chooses a correct length scale. Figure 6.6 shows the evolution of the Energy-HSIC computed on the calibration dataset over a range of one hundred length scales, from* 0.01 *to* 1*. Next, Figures 6.7a and 6.7b show the prediction bands for the chosen length scale for Liang's model and Fan's model with* $p = 2$ *respectively.*

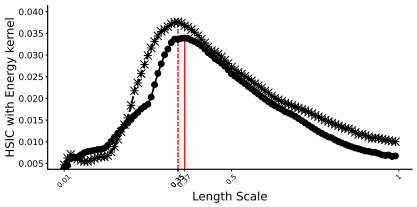
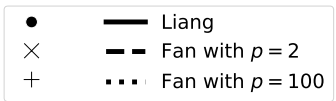
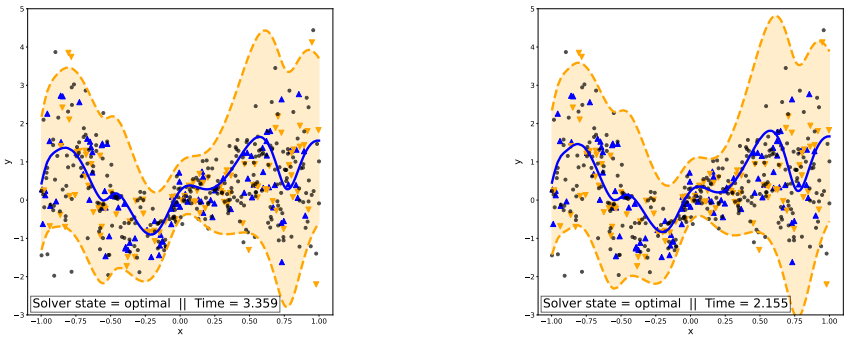


Figure 6.6: Energy-HSIC between residuals and prediction bandss width, on calibration data, on a finer scale.





(a) Liang with length scale = 0.37

(b) Fan with length scale = 0.35 and $p = 2$

Figure 6.7: Experiment 4 - Simultaneous - $\lambda_2 = 1$; $\delta = 0.001$ - Liang and Fan ($p = 2$) with the length scale chosen by the Energy-HSIC in figure Figure 6.6.

This concludes our exploration of the so called "simultaneous" model, where we estimate both the mean function and the prediction bands. Despite lacking a representer theorem, this model shows great practical results. A simultaneous estimaton of the mean function and the prediction bands seems to lead to even more adaptative bands, despite the degeneration for high length scales.

Conclusion

Safran uses machine learning models, or meta-models, to replace heavy numerical simulation algorithms. In such a high stake and precise industry, the engineers using those meta-models must have a rigorous knowledge of its uncertainty. This is where this internship took place.

Usual methods in machine learning such as random forests, Gaussian processes and Bayesian neural networks, cannot quantify the uncertainty in their prediction accurately. Furthermore, conformal prediction, a mathematically strong theory about uncertainty quantification in the predictions, presents two weaknesses. First, the exchangeability assumption is not met in Safran usecases. Second, prediction bands constructed using conformal methods are not adaptative enough.

In this thesis, we proposed to explore an alternative model to conformal prediction, based on kernel methods. First introduced by T. Liang (2022) and later developed by Fan et al. (2024), this method relies on a novel representer theorem presented by Marteau-Ferey et al. (2020). This representer theorem makes the kernel model tractable, using semi-definite programming solvers. The study of those propositions shed the light on two major problems. The first one was the calibration algorithm that, despite having theoretical foundation, did not produce desirable prediction bands. The second one was the lack of method to choose the length scale.

After having analysed this new kernel method, we proposed several improvements. First of all, we propose to leverage two results brought by Marteau-Ferey et al. (2020) to reduce the computation time by first using another formulation of the representer theorem, and second using the Frobenius regularisation. Secondly, we proposed a refined calibration procedure with the same mathematical guarantees and easy to tune parameters. Thirdly, we brought forward a metric to help choose the best length scale for estimating the variance function. Finally, we studied empirically a modified model that estimates both the mean function and the variance function simultaneously. Throughout this work, we have endeavoured to provide clear explanation of the kernel model, presenting rigorous notation for the prediction bands and for the coverage guarantee theorems, in order to compare this new method eye to eye to existing methods. We also tried to explore all hyperparameters behaviour and impact on the model.

We believe this framework opens the door for better uncertainty quantification for the predictions of machine learning models, and is suitable to adapt to Safran usecases. We are excited to pursue this work in the coming years.

Prospects

The new model investigated during this internship to estimate prediction bands is encouraging. However a lot more aspects need to be investigated in order to make it a competitive framework.

Exploration. First, we would like to explore the behaviour of this model in different scenarios. What happens when the number of observations increases? What about the number of dimensions? We know that for applications within Safran, the number of features is rarely one. Finally, we know that in Safran usecases of meta-modelling computer simulations, the observations are noiseless. Does our method adapt well to a noiseless case, interpolating the training data? And does the quantification of the model uncertainty between the training points scales correctly?

Optimisation. Following that, the Frobenius regularisation did not have the expected impact. A more thorough investigation is necessary. In the worst case scenario, λ_2 becomes a new hyper-parameter to tune. On the training time, we want to investigate a primal formulation suggested by Marteau-Ferey et al. (2020), for the sequential model. Another lead to improve training times would be to look at other semi-definite program solvers.

Hyperparameter tuning. The choice of the length scale in the Matérn kernel is hard. Investigating the behaviour of the two metrics, mutual information and Energy-HSIC, on multiple factors such as training seed and calibration seed, seems unavoidable.

Test conditional. A lot of recent work as been made lately to construct prediction bands that have test-conditional coverage. We believe the framework we just discussed in this manuscript has potential to aim for test-conditional guarantees.

Too many samples. All the models we presented here used at least three different samples. Leveraging some recent jackknife results, of Steinberger et al. (2016) and Steinberger et al. (2022), we hope to reduce the number of necessary samples.

Beyond the iid assumption. In meta-modelling for computer simulation, the experiments used to train the meta-model are often chosen using design of experiment methods. Thus the theorems supporting the train-conditional coverage do not apply. However, Fan et al. (2024) showed convergence properties of the estimator to the true variance. We believe this could be leveraged to transport the theorems to some design of experiments methods.

Simultaneous model. First, we know we can inspect the impact of values λ_2 on the training time and prediction bands shape. This model has shown really good performances on our experiment cases. Nevertheless our two Conjectures 1 and 2 are only conjectures. In this model, we used a "workaround" to estimate the mean length scale. Could we come up with an estimation that does not involve yet another sample? How does the variance metric (MI and E-HSIC) hold up in this environment when the number of dimensions goes up? With two anisotropic kernels, the number of length scales to choose is going to skyrocket... If and only if the primal formulation works for the sequential model, can it be applied here? In fine, most of the discussions we have on the sequential model, will also apply to the simultaneous model too.

Bibliography

Amann, Nicolai et al. (2023). *Assumption-lean conditional predictive inference via the Jackknife and the Jackknife+*. arXiv: 2312.14596 [math.ST]. URL: https://arxiv.org/abs/2312.14596.

Barber, Rina Foygel et al. (2020a). *Predictive inference with the jackknife+*. arXiv: 1905.02928 [stat.ME]. URL: https://arxiv.org/abs/1905.02928.

Barber, Rina Foygel et al. (2020b). *The limits of distribution-free conditional predictive inference*. arXiv: 1903.04684 [math.ST]. URL: https://arxiv.org/abs/1903.04684.

Bian, Michael et al. (2023). *Training-conditional coverage for distribution-free predictive inference*. arXiv: 2205.03647 [math.ST]. URL: https://arxiv.org/abs/2205.03647.

Breiman, L (1996). "Bagging Predictors". In: *Machine Learning* 24, pp. 123–140. URL: https://api.semanticscholar.org/CorpusID:47328136.

— (Oct. 2001). "Random Forests". In: *Machine Learning* 45, pp. 5–32. DOI: 10.1023/A:1010950718922.

Burnaev, Evgeny et al. (2014). *Efficiency of conformalized ridge regression*. arXiv: 1404.2083 [cs.LG]. URL: https://arxiv.org/abs/1404.2083.

Chung, Youngseog et al. (2021). *Beyond Pinball Loss: Quantile Methods for Calibrated Uncertainty Quantification*. arXiv: 2011.09588 [cs.LG]. URL: https://arxiv.org/abs/2011.09588.

Fan, Jianqing et al. (2024). *UTOPIA: Universally Trainable Optimal Prediction Intervals Aggregation*. arXiv: 2306.16549 [stat.ME]. URL: https://arxiv.org/abs/2306.16549.

Gammerman, Alex et al. (1998). *Learning by Transduction*. DOI: 10.48550/arXiv.1301.7375.

Genton, Marc G. (Mar. 2002). "Classes of kernels for machine learning: a statistics perspective". In: *J. Mach. Learn. Res.* 2, pp. 299–312. ISSN: 1532-4435.

Gretton, Arthur et al. (2005). "Measuring Statistical Dependence with Hilbert-Schmidt Norms". In: *Algorithmic Learning Theory*. Ed. by Sanjay Jain et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 63–77. ISBN: 978-3-540-31696-1.

Gretton, Arthur et al. (2012). "A Kernel Two-Sample Test". In: *Journal of Machine Learning Research* 13.25, pp. 723–773. URL: http://jmlr.org/papers/v13/gretton12a.html.

Jaber, Edgar et al. (2024). *Conformal Approach To Gaussian Process Surrogate Evaluation With Coverage Guarantees*. arXiv: 2401.07733 [stat.ML]. URL: https://arxiv.org/abs/2401.07733.

Johansson, Ulf et al. (Oct. 2014). "Regression conformal prediction with random forests". In: *Mach. Learn.*, pp. 155–176. ISSN: 0885-6125. DOI: 10.1007/s10994-014-5453-0. URL: https://doi.org/10.1007/s10994-014-5453-0.

Kraskov, Alexander et al. (June 2004). "Estimating mutual information". In: *Phys. Rev. E* 69 (6), p. 066138. DOI: 10.1103/PhysRevE.69.066138. URL: https://link.aps.org/doi/10.1103/PhysRevE.69.066138.

Krige, Daniel G. (1951). "A statistical approach to some basic mine valuation problems on the Witwatersrand, by D.G. Krige, published in the Journal, December 1951 : introduction by the author". In: *Journal of The South African Institute of Mining and Metallurgy* 52, pp. 201–203. URL: https://api.semanticscholar.org/CorpusID:117319434.

Lei, Jing (2017). *Fast Exact Conformalization of Lasso using Piecewise Linear Homotopy*. arXiv: 1708.00427 [stat.ME]. URL: https://arxiv.org/abs/1708.00427.

Lei, Jing et al. (2017). *Distribution-Free Predictive Inference For Regression*. arXiv: 1604.04173 [stat.ME]. URL: https://arxiv.org/abs/1604.04173.

Liang, Ruiting et al. (2024). *Algorithmic stability implies training-conditional coverage for distribution-free prediction methods*. arXiv: 2311.04295 [math.ST]. URL: https://arxiv.org/abs/2311.04295.

Liang, Tengyuan (Aug. 2022). "Universal Prediction Band via Semi-Definite Programming". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 84.4, pp. 1558–1580. ISSN: 1467-9868. DOI: 10.1111/rssb.12542. URL: http://dx.doi.org/10.1111/rssb.12542.

Marteau-Ferey, Ulysse et al. (2020). "Non-parametric Models for Non-negative Functions". In: *CoRR* abs/2007.03926. arXiv: 2007.03926. URL: https://arxiv.org/abs/2007.03926.

Matérn, Bertil (1960). *Spatial variation*. Vol. 36. Springer Science & Business Media.

Matheron, G. (1962). *Traité de géostatistique appliquée*. Memoires vol. 1. Éditions Technip. URL: https://books.google.fr/books?id=88YKAQAAMAAJ.

Meinshausen, Nicolai et al. (2006). "Quantile regression forests." In: *Journal of machine learning research* 7.6.

O'Donoghue, Brendan et al. (June 2016). "Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding". In: *Journal of Optimization Theory and Applications* 169.3, pp. 1042–1068. URL: http://stanford.edu/~boyd/papers/scs.html.

— (Oct. 2023). *SCS: Splitting Conic Solver, version 3.2.7*. https://github.com/cvxgrp/scs. URL: https://github.com/cvxgrp/scs.

Papadopoulos, H. et al. (Apr. 2011). "Regression Conformal Prediction with Nearest Neighbours". In: *Journal of Artificial Intelligence Research* 40, pp. 815–840. ISSN: 1076-9757. DOI: 10.1613/jair.3198. URL: http://dx.doi.org/10.1613/jair.3198.

Papadopoulos, Harris (2023). *Guaranteed Coverage Prediction Intervals with Gaussian Process Regression*. arXiv: 2310.15641 [cs.LG]. URL: https://arxiv.org/abs/2310.15641.

Papadopoulos, Harris et al. (2002). "Inductive Confidence Machines for Regression". In: *Machine Learning: ECML 2002*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 345–356. DOI: 10.1007/3-540-36755-1_29.

Pion, Aurélien et al. (2024). *Gaussian process interpolation with conformal prediction: methods and comparative analysis*. arXiv: 2407.08271 [cs.LG]. URL: https://arxiv.org/abs/2407.08271.

Rasmussen, Carl Edward (2004). "Gaussian Processes in Machine Learning". In: *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany,*

August 4 - 16, 2003, Revised Lectures. Ed. by Olivier Bousquet et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 63–71. ISBN: 978-3-540-28650-9. DOI: 10.1007/978-3-540-28650-9_4. URL: https://doi.org/10.1007/978-3-540-28650-9_4.

Rasmussen, Carl Edward et al. (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, pp. I–XVIII, 1–248. ISBN: 026218253X. URL: https://gaussianprocess.org/gpml/chapters/RW.pdf.

Romano, Yaniv et al. (2019). *Conformalized Quantile Regression*. arXiv: 1905.03222 [stat.ME]. URL: https://arxiv.org/abs/1905.03222.

Steinberger, Lukas et al. (2016). *Leave-one-out prediction intervals in linear regression models with many variables*. arXiv: 1602.05801 [math.ST]. URL: https://arxiv.org/abs/1602.05801.

— (2022). *Conditional predictive inference for stable algorithms*. arXiv: 1809.01412 [math.ST]. URL: https://arxiv.org/abs/1809.01412.

Tibshirani, Ryan (2024). *Conformal Prediction*. URL: https://www.stat.berkeley.edu/~ryantibs/statlearn-s24/.

Vovk, Vladimir (2012a). *Conditional validity of inductive conformal predictors*. arXiv: 1209.2673 [cs.LG]. URL: https://arxiv.org/abs/1209.2673.

— (2012b). *Cross-conformal predictors*. arXiv: 1208.0806 [stat.ML]. URL: https://arxiv.org/abs/1208.0806.

Vovk, Vladimir et al. (Jan. 2005). *Algorithmic Learning in a Random World*. Springer, Berlin, Heidelberg. DOI: 10.1007/b106715.

Vovk, Vladimir et al. (2019). *Computationally efficient versions of conformal predictive distributions*. arXiv: 1911.00941 [cs.LG]. URL: https://arxiv.org/abs/1911.00941.

Appendix A

Experiments description

We are going to consider four toy datasets to evaluate and compare the different algorithms get study here. Here we present the data generation process for all experiments and provide a figure of a one thousand data points sample, with the true mean function in red and the true 0.95 and 0.05 quantile functions in green.

Experiment 1. The first example is very simple. We consider a null mean function and a constant variance with a white noise (as in taken from a $0, 1$ normal distribution).

$$x \sim \mathcal{U}(-2, 2)$$

$$\varepsilon \sim \mathcal{N}(0, 1)$$

$$v(x) = 1$$

$$y|x = \varepsilon \cdot \sqrt{v(x)}$$

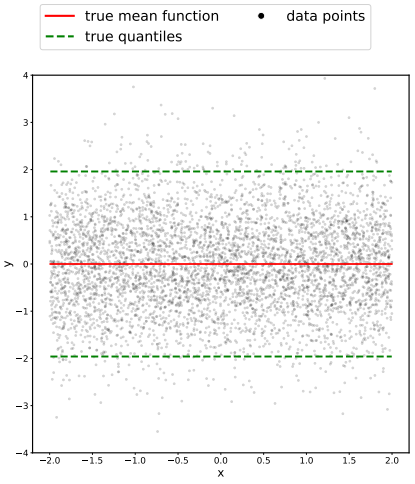


Figure A.1: First toy dataset.

Experiment 2. The second example is similar to the first one but with a quadratic variance function.

$$x \sim \mathcal{U}(-2, 2)$$

$$\varepsilon \sim \mathcal{N}(0, 1)$$

$$v(x) = 1 + x + 4x^2$$

$$y|x = \varepsilon \cdot \sqrt{v(x)}$$

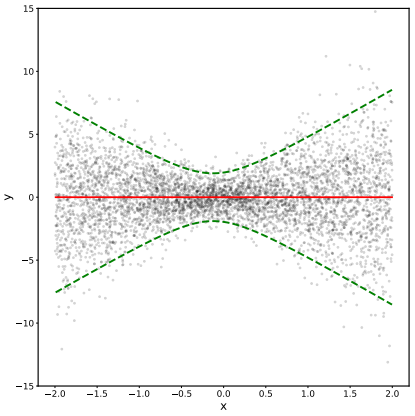


Figure A.2: Second toy dataset.

Experiment 3. The third example is similar to the first two ones but with a variance function that is to the power four.

$$x \sim \mathcal{U}(-2, 2)$$

$$\varepsilon \sim \mathcal{N}(0, 1)$$

$$v(x) = 1 + 4x^4$$

$$y|x = \varepsilon \cdot \sqrt{v(x)}$$

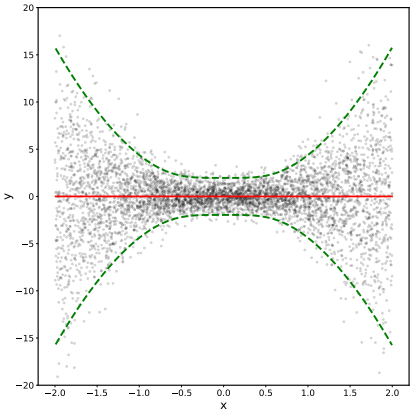


Figure A.3: Third toy dataset.

Experiment 4. Finally, our fourth experiment is strictly different from the first three. First of all because the mean function is no longer null. Furthermore, we have a non derivable point at $x = -0.04$.

$$x \sim \mathcal{U}(-1, 1)$$
$$\varepsilon \sim \mathcal{N}(0, 1)$$
$$v(x) = g(10(1 + x))$$
$$vn(x) = 0.1 + 2x^2$$
$$y|x = v(x) + \varepsilon\sqrt{vn(x)}$$

Where g is the Gramacy function: $g(x) = \begin{cases} \sin\left(\frac{2\pi}{5}\right) + \frac{1}{5}\cos\left(\frac{4\pi x}{5}\right) & x \leq 9.6 \\ \frac{x}{10} - 1 & 9.6 < x \end{cases}$

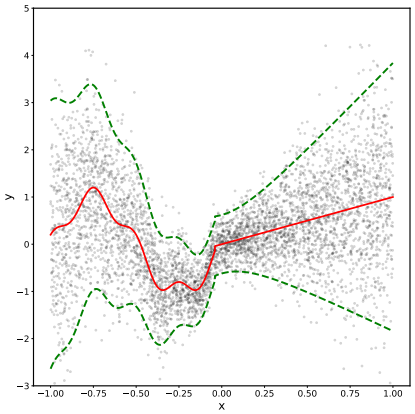


Figure A.4: Fourth toy dataset.



This fourth experiment will be our *red thread* along this manuscript. We will showcase most of the methods discussed with this example. In our figures we will adopt the following colour code. Blue will be attributed to the training dataset and to the estimated mean function. Orange will be for the calibration dataset (when there is one) and for the estimation of the prediction bands.

Finally, in most of our discussions on empirical studies, we will focus on experiment 2 and 4. The first three experiments are really close to each other, with a null mean function, therefore the discussions are often similar.

Fixed variables. Because our experiments involve numerous variables which are all important and interesting to test, we decided to fix some of them in order to simplify our work and focus on the variables that really matter. The first variable we fixed is the kernel used in all models that leverages a kernel. We decided to use the Matérn 5/2, introduced by Matérn (1960), as it is one of the most versatile out there. A good review of kernels can be found in Genton (2002). It is a special case of the Matérn kernel, with smoothness parameter ν equal to 5/2.

Definition 10 (Matérn 5/2 kernel)**.** *Let $\mathcal{X} \subset \mathcal{H}$, $\mathcal{H}$ a Hilbert space. Let $l > 0$ be the length scale, the Matérn 5/2 kernel $k_{l,\frac{5}{2}} : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is given by:*

$$k_{l,\frac{5}{2}}(x, x') = \left(1 + \frac{\sqrt{5}||x - x'||_{\mathcal{H}}}{l} + \frac{5||x - x'||^2_{\mathcal{H}}}{3l^2}\right) \exp\left(-\frac{5||x - x'||_{\mathcal{H}}}{l}\right)$$

The second variable we decided to fix is the confidence level of the prediction bands. Throughout all the experiments we will target a 0.95 coverage. Thus α is fixed to 0.05, and we will reserve this character for the coverage level. Third, in the experiments we are going to use a semi-definite program solver, the Splitting Conic Solver (SCS) algorithm, presented in O'Donoghue et al. (2016). It is open source and available at O'Donoghue et al. (2023). Lastly, a profiling of SCS training time showed that most of the time was spent on checking that our kernel matrix was indeed positive-definite. Therefore we disabled all positive-definite checks during training. The times presented in this thesis are all without numerically checking that the kernel matrix is positive definite.

Appendix B

All figures details

Throughtout all our report we used the experiment four as a red thread to illustrate most methods. Not all methods use a calibration dataset, some need to estimate the mean function on a separate dataset. In this appendix we will discuss precisely which data were used for what purpose. Of course, splitting a dataset is a hyper-parameter in itself. We will not discuss this here and split the dataset in equal parts, when the dataset needs to be split.

In total we considered four datasets, the first three of size 100 and the last one, the test dataset, of size 300:

$$\begin{aligned} &\text{i) } \mathcal{D}_p\text{: pre-train dataset,} && \text{ii) } \mathcal{D}_n\text{: training dataset,} \\ &\text{iii) } \mathcal{D}_m\text{: calibration dataset,} && \text{iv) } \mathcal{D}_t\text{: test dataset.} \end{aligned} \tag{B.1}$$

Kernel methods, sequential. In all kernel methods that used a sequential estimation of the mean function and the prediction bands, which means all prediction bands in chapters 4 and 5, we used the pre-train dataset to estimate the mean function (using a Gaussian process model) $\hat{m}_p$. Then we used the train dataset to learn the variance function $\hat{v}_n$. And finally we used the calibration dataset to estimate the coefficient $\hat{\lambda}_m$ or $\hat{\delta}_m$, depending on the calibration algorithm used.

Kernel methods, simultaneous. For the simultaneous kernel model, we used the pre-train dataset to train a Gaussian process in order to estimate some hyper-parameters of the kernel model. Then we used the training dataset to train both $\hat{m}_n$ and $\hat{v}_n$. Finally, we used the calibration dataset to estimate $\hat{\lambda}_m$.

Traditional methods: Gaussian Processes. For the model that used only a Gaussian process we used the reunion of the three datasets to train the model. The training data for the Gaussian process was thus: $\mathcal{D}_p \cup \mathcal{D}_n \cup \mathcal{D}_m$.

Traditional methods: Random Forests. Same as for the Gaussian process, we used a reunion of the three datasets to train the model: $\mathcal{D}_p \cup \mathcal{D}_n \cup \mathcal{D}_m$

Traditional methods: Kernel Quantile Regression. The problem with this method is that we need to choose the hyper-parameters of the kernel. To deal with this, we decided to first train a Gaussian process on the pre-train dataset, computing $\hat{m}_p$. We then used the posterior kernel for

the quantile regression kernel and trained the upper and lower quantiles on $\mathcal{D}_n \cup \mathcal{D}_m$.

Conformal prediction: Split. For the split conformal prediction model, we used a Gaussian process underneath. We trained the Gaussian process on the pre-trained dataset and half of the training dataset: $\mathcal{D}_p \cup \mathcal{D}_{1:\frac{n}{2}}$. Then, we used the second half of the training dataset and combined with the calibration dataset, $\mathcal{D}_{\frac{n}{2}:n} \cup \mathcal{D}_m$, to compute the finite sample correction quantile.

Conformal prediction: Jackknife+ and J+GP. Similarly to Gaussian processes and random forests we used the combination of all three datasets to train those models: $\mathcal{D}_p \cup \mathcal{D}_n \cup \mathcal{D}_m$.

Conformal prediction: Conformalized Quantile Regression. We faced the same problem for the hyper-parameters of the kernels here, as in the kernel quantile regression model. Therefore we chose to use the pre-train dataset to train a Gaussian process and estimate the mean function $\hat{m}_p$ and estimate the hyper-parameters for the kernel used in the quantile regression part. The lower and upper quantiles were trained on the training dataset $\mathcal{D}_n$. Finally, we computed the scores and the finite sample correction quantile on the calibration dataset $\mathcal{D}_m$.

Appendix C

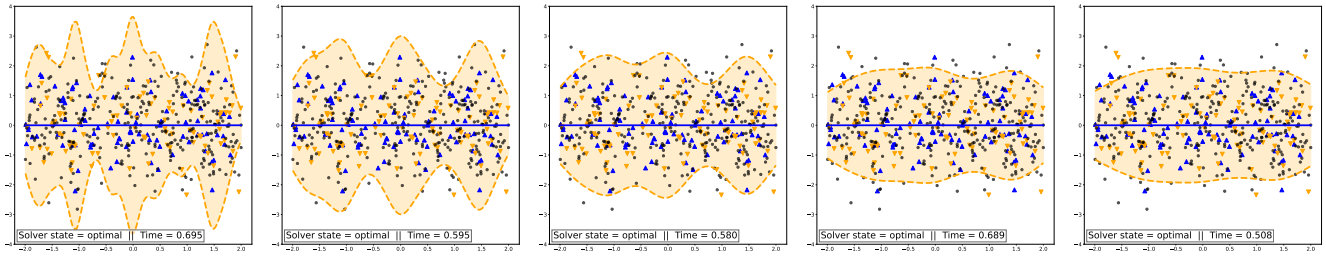
Experiments complementary results

In the exploration of experiments one and three, we are not going to study Liang's and Fan's model with formulation B, as we understood that the calibration procedures were not reliale. Thus, we are going to use formulation A presented in chapter 5 as well as the calibration procedure presented in the same chapter.

Experiment 1

Sequential model

In figure C.1 we present Liang's objective function and in figure C.2 we showcase Fan's objective function with hyper-parameter $p = 2$ in the top row and $p = 100$ in the bottom row. Per usual, the mean function has been estimated using au Gaussian process model on a different dataset.

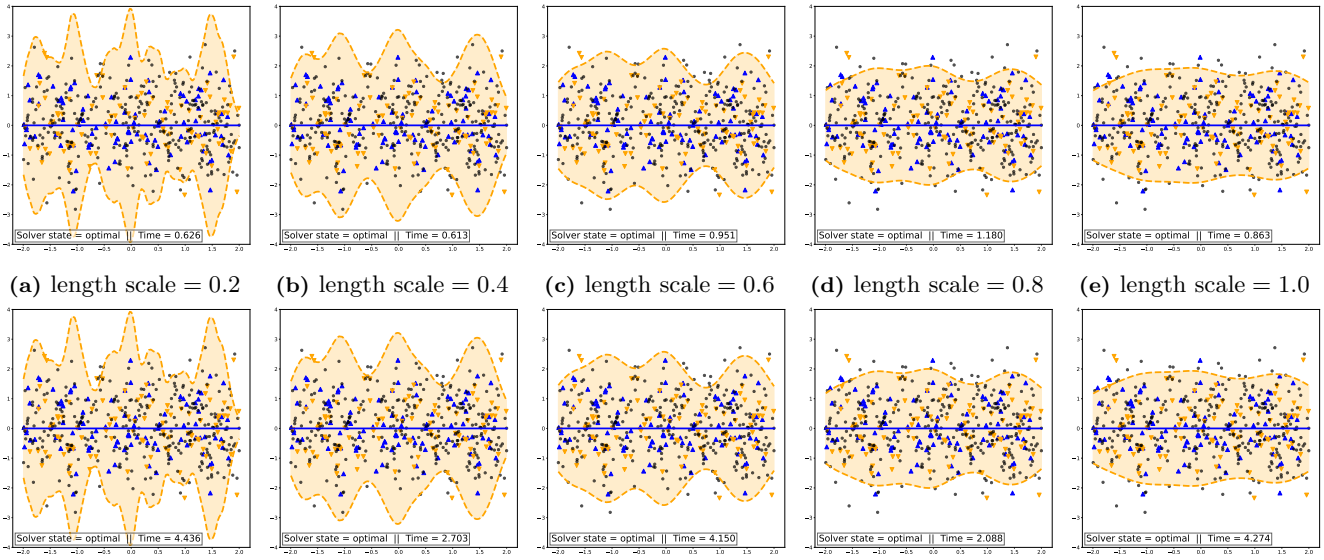


(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure C.1: Experiment 1 - Liang A - $\lambda_2 = 1; \delta = 0.001$ - Evolution of the mean function and prediction bands' shape with regard to the length scale.

Simultaneous model

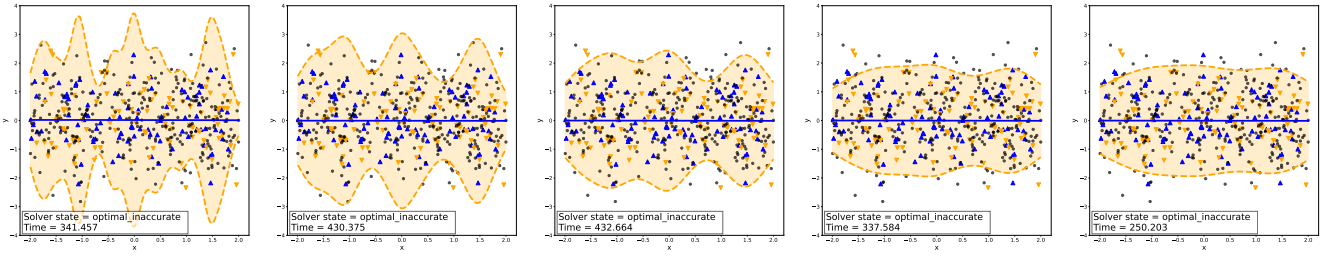
In figure C.3 we present the simultaneous model using Liang's objective function and in figure C.4 we present the simultaneous model with Fan's objective function. The estimation of the hyper-parameters s and the length scale of the mean function have been estimated using a Gaussian process model trained on a different dataset.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

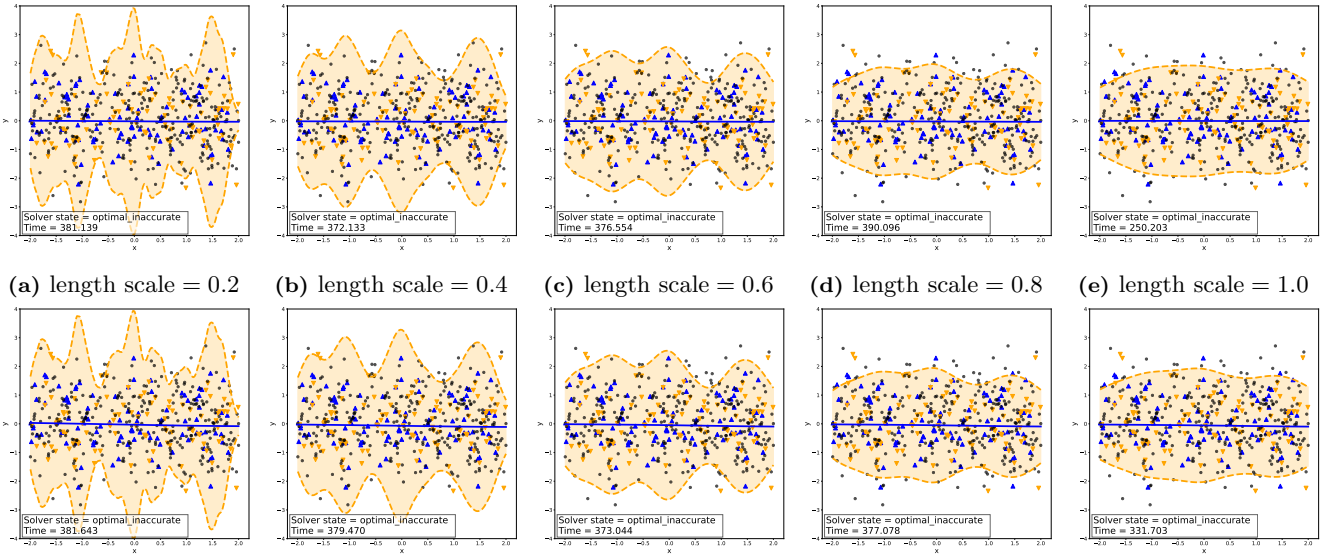
(f) length scale = 0.2 **(g)** length scale = 0.4 **(h)** length scale = 0.6 **(i)** length scale = 0.8 **(j)** length scale = 1.0

Figure C.2: Experiment 1 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$ - Evolution of the mean function and the prediction bands' shape with regard to the length scale.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure C.3: Experiment 1 - Liang A, simultaneous - $\lambda_2 = 1$;$\delta = 0.001$ - Evolution of the mean function and prediction bands' shape with regard to the length scale.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

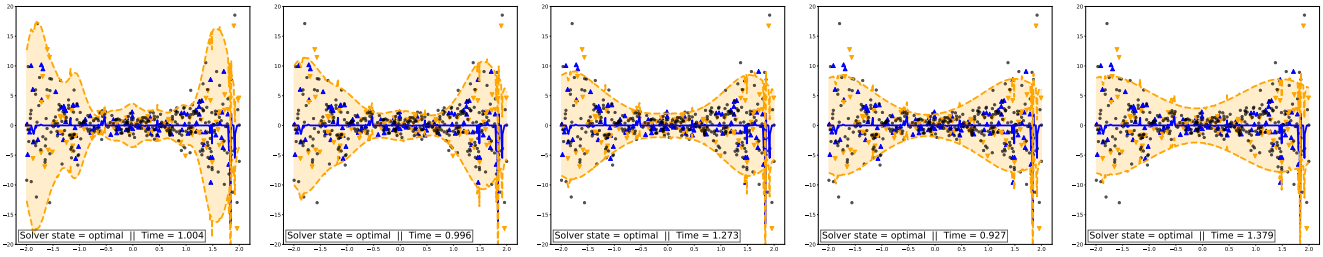
(f) length scale = 0.2 **(g)** length scale = 0.4 **(h)** length scale = 0.6 **(i)** length scale = 0.8 **(j)** length scale = 1.0

Figure C.4: Experiment 1 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$ - Evolution of the mean function and the prediction bands' shape with regard to the length scale.

Experiment 3

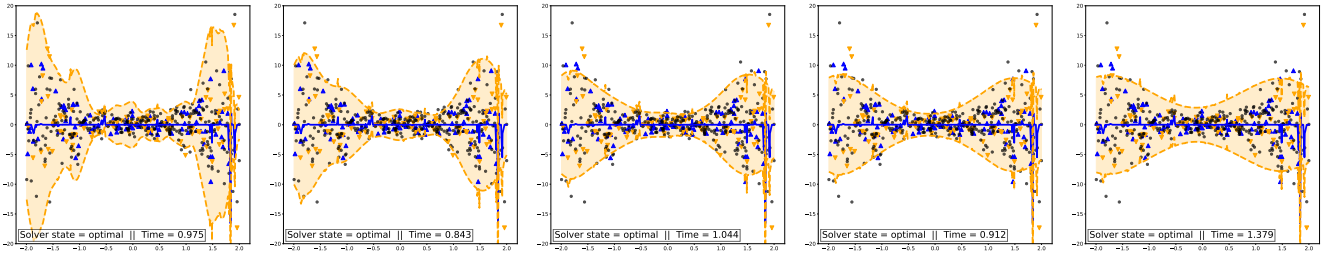
Sequential model

In figure C.5 we present Liang's objective function and in figure C.6 we showcase Fan's objective function with hyper-parameter $p = 2$ in the top row and $p = 100$ in the bottom row. Per usual, the mean function has been estimated using au Gaussian process model on a different dataset.

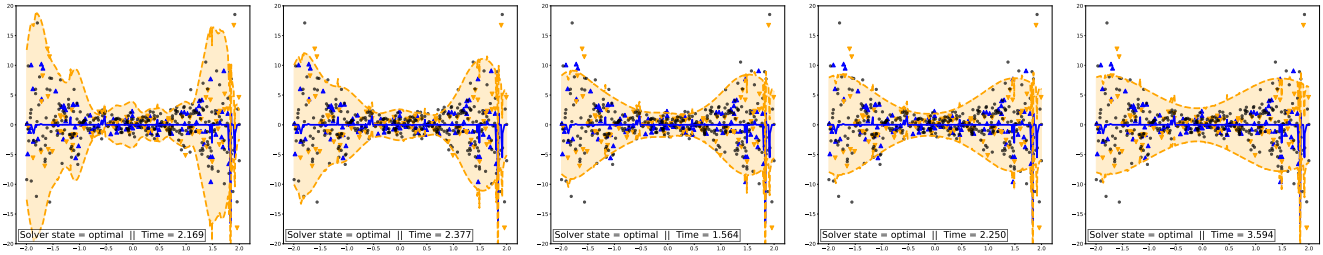


(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure C.5: Experiment 3 - Liang A - $\lambda_2 = 1; \delta = 0.001$ - Evolution of the mean function and prediction bands' shape with regard to the length scale.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

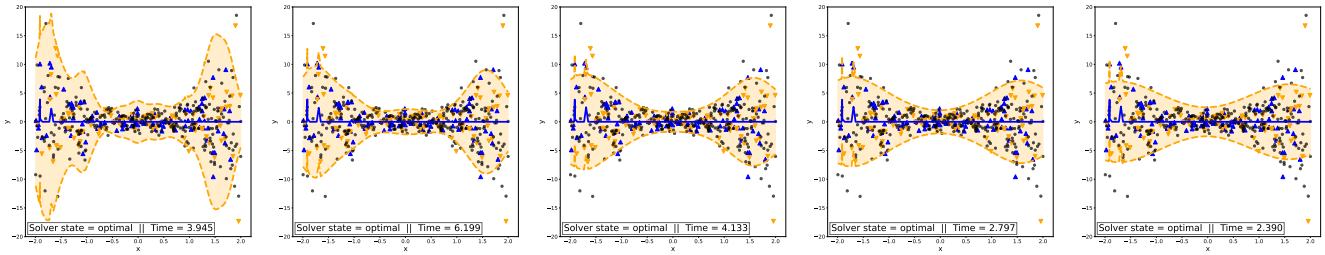


(f) length scale = 0.2 **(g)** length scale = 0.4 **(h)** length scale = 0.6 **(i)** length scale = 0.8 **(j)** length scale = 1.0

Figure C.6: Experiment 3 - Fan A - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2, 100$ - Evolution of the mean function and the prediction bands' shape with regard to the length scale.

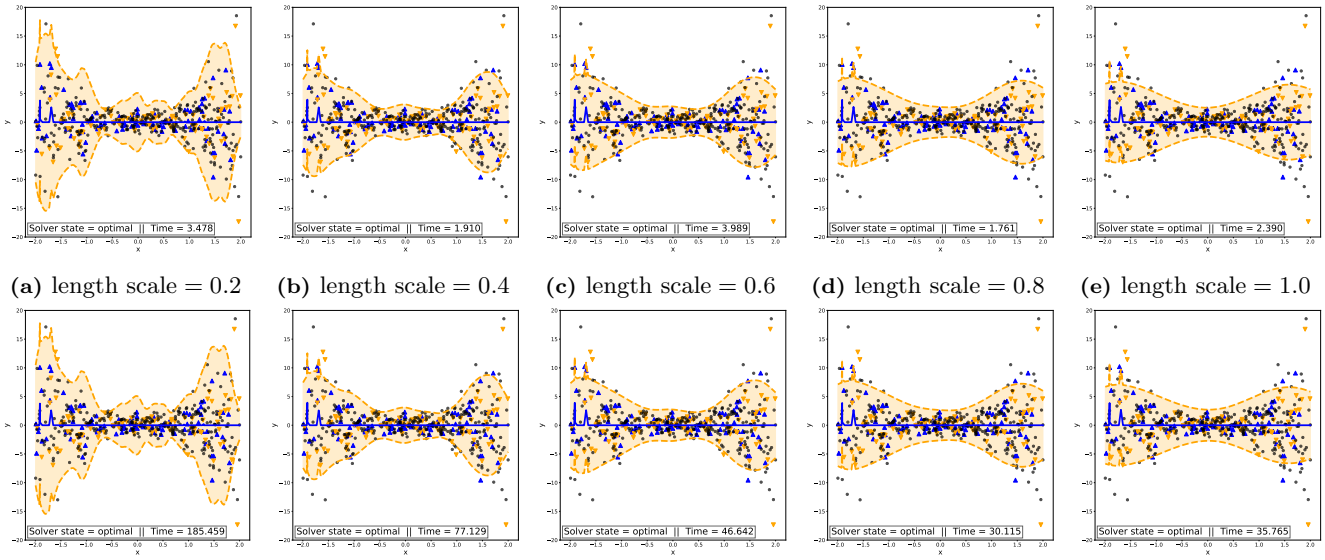
Simultaneous model

In figure C.7 we showcase the simultaneous model with Liang's objective function and in figure C.8 we present the simultaneous model with Fan's objective function. The hyper-parameters s and the length scale of the mean function have been estimated by training a Gaussian process on a different dataset.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure C.7: Experiment 3 - Liang A, simultaneous - $\lambda_2 = 1; \delta = 0.001$ - Evolution of the mean function and prediction bands' shape with regard to the length scale.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

(f) length scale = 0.2 **(g)** length scale = 0.4 **(h)** length scale = 0.6 **(i)** length scale = 0.8 **(j)** length scale = 1.0

Figure C.8: Experiment 3 - Fan A, simultaneous - $\lambda_2 = 1$; $\delta = 0.001$; $p = 2,100$ - Evolution of the mean function and the prediction bands' shape with regard to the length scale.

Appendix D

Conformal predictions

This theory has emerged as a strong mathematically and practically reliable method for uncertainty quantification in the predictions. It originated from Gammerman et al. (1998) and since then numerous propositions were made to improve the method. We heavily rely on the lecture notes from Tibshirani (2024) to present conformal predictions. Other important sources are Vovk et al. (2005) and Lei et al. (2017). The theory on conformal prediction relies on the prediction quality of a machine learning model to asses the uncertainty of this prediction, thus yielding a prediction interval rather than a point prediction.

Conformal prediction aims to quantify the degree of certainty one has over a prediction of a machine learning model. Formally, for a given error rate $\alpha \in]0,1[$, we want to train a prediction set $\widehat{C}_n \colon \mathcal{X} \to \text{subset}(\mathcal{Y})$ such that for any new observation $(X_{\text{new}}, Y_{\text{new}})$ we have:

$$\mathbb{P}\left(Y_{\text{new}} \in \widehat{C}_n(X_{\text{new}})\right) \geq 1 - \alpha \tag{D.1}$$

That is the goal of conformal prediction. With that, we also want our model to have a few distinct features that will set conformal prediction apart from other models. First, we want 3.1 to hold in *finite sample*. Second, we want the model to be *distribution free*, we do not want any assumptions on the real distribution of the data. Third, we want to have *adaptability* of the prediction bands, meaning we do not want $\widehat{C}_n$ to be of constant width. Fourth, we would prefer a model that is *agnostic* to the machine learning point predictor. Indeed we would like to construct such set around a machine learning model. We will now develop on the finite sample correction which is key in conformal prediction.

D.1 A naive idea

The very first idea one can have to verify 3.1 is to compute the $1-\alpha$ quantile. Lets see on a simpler case why this idea is too simple. Consider a sequence of observations $S_i \in \mathbb{R}$, $i = 1, \ldots, n$, a test point S_{n+1} and a one sided prediction set $\widehat{C}_n =]-\infty, \hat{q}_n]$. Our objective 3.1 becomes

$$\mathbb{P}\left(S_{n+1} \in \widehat{C}_n(X_{n+1})\right) = \mathbb{P}\left(S_{n+1} \leq \hat{q}_n\right) \geq 1 - \alpha \tag{D.2}$$

Recall the definition of a quantile and by choosing $\hat{q}_n$ as the $1-\alpha$ empirical quantile of the $S_i, i = 1, \ldots, n$,

$$\hat{q}_n = Q\left(1 - \alpha, \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}\right) \tag{D.3}$$

we obtain the approximate result $\mathbb{P}(S_{n+1} \leq \hat{q}_n) \approx 1-\alpha$, which becomes exact as n tends to infinity

$$\text{for } n \to \infty, \mathbb{P}(Y_{n+1} \leq \hat{q}_n) \geq 1-\alpha \tag{D.4}$$

Our objective here is to obtain the same guarantee in finite sample, we want it to be true for all n and not only when $n \to \infty$.

To achieve this we first need to assume that our observations $Y_1, \ldots, Y_{n+1}$ are exchangeable[23].

[23] Exchangeability is weaker than independent and identically distributed (iid). If a sample is iid than it is exchangeable.

Definition 11 (Exchangeability)**.** *$S_1, \ldots, S_{n+1}$ are said to be exchangeable if their joint distribution is invariant under permutation:*

$$\forall \sigma \in \mathfrak{S}, (S_1, \ldots, S_{n+1}) \overset{\mathrm{d}}{=} \left(S_{\sigma(1)}, \ldots, S_{\sigma(n+1)}\right) \tag{D.5}$$

Because our observations are exchangeable, the rank of Y_{n+1} is uniformly distributed over the ranks of $S_1, \ldots, S_{n+1}$. S_{n+1} is equally likely to occupy any place in the ranked observations. Therefore, by definition of the $1-\alpha$ empirical quantile (of a $n+1$ sample) $S_{\lceil (1-\alpha)(n+1) \rceil}$, this is equivalent to:

$$\mathbb{P}\left(S_{n+1} \leq S_{\lceil (1-\alpha)(n+1) \rceil}\right) \geq 1-\alpha \tag{D.6}$$

which can be translated to the probability of S_{n+1} being among the $\lceil (1-\alpha)(n+1) \rceil^{\text{th}}$ smallest value of $S_1, \ldots, S_{n+1}$ is greater than $1-\alpha$.

$$\underbrace{S_{(1)}, \ldots, S_{\lceil (1-\alpha)(n+1) \rceil}}_{\text{probability of } S_{n+1} \text{ being here is greater than } 1-\alpha}, \ldots, S_{(n+1)}$$

Now, consider the event

$$A \coloneqq (S_{n+1} \text{ being among the } \lceil (1-\alpha)(n+1) \rceil \text{ smallest } S_1, \ldots S_{n+1})$$

We have

$$\bar{A} = (S_{n+1} > \lceil (1-\alpha)(n+1) \rceil \text{ smallest } S_1, \ldots S_{n+1})$$

Because $S_{n+1} > S_{n+1}$ is impossible,

$$\begin{aligned} &(S_{n+1} > \lceil (1-\alpha)(n+1) \rceil \text{ smallest } S_1, \ldots S_{n+1}) \\ =&(S_{n+1} > \lceil (1-\alpha)(n+1) \rceil \text{ smallest } S_1, \ldots S_n) \end{aligned}$$

Going back up the equations, we obtain

$$A = (S_{n+1} \text{ being among the } \lceil (1-\alpha)(n+1) \rceil \text{ smallest } S_1, \ldots S_n)$$

And thus the probability of S_{n+1} being among the $\lceil (1-\alpha)(n+1) \rceil^{\text{th}}$ smallest value of $S_1, \ldots, S_n$ is greater than $1-\alpha$. Now, if we write[24]

$$\hat{q}_n \coloneqq \lceil (1-\alpha)(n+1) \rceil \text{ smallest of } S_1, \ldots, S_n$$

we achieve $\mathbb{P}(S_{n+1} \leq \hat{q}_n) \geq 1-\alpha$ in finite sample.

[24] This is equivalent to defining $\hat{q}_n = Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}\right)$.

The newly defined $\hat{q}_n$ can be seen as the empirical quantile at an adjusted level to allow for a finite sample guarantee, this is the *finite sample correction*. This adjustment will be key in the theoretical guarantees of conformal prediction methods.

D.2 Full conformal prediction

Now we want to lift this idea and use it in a regression setting. The very first case of conformal prediction is the one called full or transductive conformal prediction. It is the oldest model, proposed by Gammerman et al. (1998). Let $\mathcal{D}_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$ an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(X_i, Y_i) \sim P$ where P is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(X_{n+1}, Y_{n+1}) \sim P$ be a test point. To construct the prediction set $\widehat{C}_n(X_{n+1})$ we are going to apply the finite sample correction to some residuals. In order for them to be exchangeable we need the residuals to be constructed in a specific way.

Let $y \in \mathcal{Y}$ a "trial value" for our test point X_{n+1}. Consider the augmented training dataset $\mathcal{D}_a = \{(X_1, Y_1), \dots, (X_n, Y_n), (X_{n+1}, y)\}$ and train a point predictor regression function on $\mathcal{D}_a$:

$$\hat{f}_{\mathcal{D}_a} \colon \mathcal{X} \to \mathcal{Y}$$

Then, compute the residuals

$$R_i^{(X_{n+1}, y)} = |Y_i - \hat{f}_{\mathcal{D}_a}(X_i)|, \ i \in [\![1, n]\!]$$

Next, the prediction set for X_{n+1} can be constructed using the finite sample correction:

$$\widehat{C}_n(X_{n+1}) = \left\{ y \colon R_{n+1}^{(X_{n+1}, y)} \leq Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n} \sum_{i=1}^{n} \delta_{R_i^{(X_{n+1}, y)}} \right) \right\}$$

Constructing the prediction set this way gives the following theorem:

Theorem 15 (Marginal coverage for full conformal prediction)
If $(X_1, Y_1), \dots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ are exchangeable and if the prediction algorithm is symmetric, than the full conformal prediction set satisfies[25]

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1}) \right) \geq 1 - \alpha. \tag{D.7}$$

[25] The proof is given in Vovk et al. (2005)

In order to compute the prediction set for a given input X_{n+1}, the machine learning model has to be trained for every $y \in \mathcal{Y}$. In theory an infinite amount, because $\mathcal{Y} \in \mathbb{R}$ is continuous, in practice we define a grid of values to test. To predict the test set for a test sample of size n_{test} we need to compute the machine learning model $n_{\text{test}} \times n_{\text{grid}}$, which is computationally infeasible in most cases. Fortunately for some algorithm, there exists computational tricks to avoid this burden: k-nearest-neighbours (H. Papadopoulos et al. (2011)), ridge regression (Burnaev et al. (2014)) and lasso regression (Lei (2017)).

Theorem 15 proposes a marginal guarantee for coverage. This means that it holds on average over the training sample, however for a given training dataset the coverage might not be met. Very recently, R. Liang et al. (2024), proposed a new theorem on full conformal prediction to allow for training conditional coverage guarantee in exchange to added assumptions. The first supplementary assumption is about algorithm stability. What we need is for the algorithm that constructs the point predictor $\hat{f}$ to be stable (to give similar output) under a slight modification of the training data, especially when predicting for a point within the training dataset. This is called "in-sample m stability" in R. Liang et al. (2024) (definition 3.4) and denoted $\beta_{m,n}^{\text{in}}$. The second modification is in the construction of the

set $\widehat{C}_n$, they added an inflation parameter $\gamma > 0$, which plays the role of an offset:

$$\widehat{C}_n^{\gamma}(X_{n+1}) = \left\{ y \colon R_{n+1}^{(X_{n+1},y)} \leq Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n} \sum_{i=1}^{n} \delta_{R_i^{(X_{n+1},y)}} \right) + \gamma \right\}$$

With those two modifications in mind, R. Liang et al. (2024) writes:

Theorem 16 (Train-conditional coverage for full conformal prediction) *For any point prediction algorithm $\mathcal{A}$ that is symmetric, for any $\gamma > 0$ inflation parameter, for any $m \leq 1$ and any $\delta \in (0,1)$*[26]*:*

$$\mathbb{P}\left[\text{COV}\right] \geq 1 - 3\delta - \sqrt[3]{\frac{2\beta_{m-1,n+1}^{in}(\mathcal{A}, P)}{\gamma}} \tag{D.8}$$

Where COV *is the event of satisfying the coverage guarantee:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n^{\gamma}(X_{n+1}) | \mathcal{D}_n\right) \geq 1 - \alpha - 3\sqrt{\frac{\log(1/\delta)}{2\min(n,m)}} - 2\sqrt[3]{\frac{2\beta_{m-1,n+1}^{in}(\mathcal{A}, P)}{\gamma}} \tag{D.9}$$

[26] A complete overview of this theorem can be found in section 3.2 of R. Liang et al. (2024) and the proof is in section 6.3 of the same article.

Full conformal prediction sets the foundation for a strong theory of learning a set predictor regression function. Its heavy computation makes this model hardly viable in practice despite efforts to reduce computation and improve coverage guarantees. In the following sections we will explore variants methods to improve full conformal prediction performances.

D.3 Split conformal prediction

Very early on, Harris Papadopoulos et al. (2002), proposed to split the training dataset to achieve exchangeability of the residuals without the computational burden[27]. Let $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(X_i, Y_i) \sim P$ where P is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(X_{n+1}, Y_{n+1}) \sim P$ be a test point. First of all, lets *split* the training dataset into two, a proper training dataset $\mathcal{D}_1$ of size n_1 and a calibration one $\mathcal{D}_2$ of size n_2. We begin by training a point predictor regression function[28] on the proper training dataset $\mathcal{D}_1$:

$$\hat{f}_{n_1} \colon \mathcal{X} \to \mathcal{Y} \tag{D.10}$$

Then, we compute the residuals on of this model on the hold out calibration dataset $\mathcal{D}_2$:

$$R_i = |Y_i - \hat{f}_{n_1}(X_i)|, \ i \in \mathcal{D}_2 \tag{D.11}$$

After that we can apply the finite sample correction on the residuals $R_i, \ i \in \mathcal{D}_2$ to compute the adjusted quantile:

$$\hat{q}_{n_2} = Q\left(\frac{\lceil (1-\alpha)(n_2+1) \rceil}{n_2}, \frac{1}{n_2} \sum_{i=1}^{n_2} \delta_{R_i} \right) \tag{D.12}$$

Then the prediction set writes for X_{n+1}:

$$\begin{aligned} \widehat{C}_n(x) &= \{y \colon R_{n+1} \leq \hat{q}_{n_2}\} \\ &= \{y \colon |Y_{n+1} - \hat{f}_{n_1}(X_{n+1})| \leq \hat{q}_{n_2}\} \\ &= \left[\hat{f}_{n_1}(X_{n+1}) - \hat{q}_{n_2}, \ \hat{f}_{n_1}(X_{n+1}) + \hat{q}_{n_2}\right] \end{aligned} \tag{D.13}$$

This procedure gives birth to the following theorem:

[27] This method can be found under the name of inductive conformal prediction. We will stick to the machine learning name of split conformal prediction.

[28] It can be any machine learning such as kernel ridge regression, splines, ensembles methods, neural networks...

[29] The proof is given in Harris Papadopoulos et al. (2002) and Vovk et al. (2005). It is immediate by observing that the R_i are exchangeable.

Theorem 17 (Marginal coverage for split conformal prediction)
With the only assumption that $= (X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ *are exchangeable, the the split conformal prediction set satisfies*[29]

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - \alpha. \tag{D.14}$$

Remark 8. *Split conformal prediction takes its name from the fact that we split the whole dataset into two. This is mandatory. Indeed lets suppose for a moment that we do not split the dataset and consider only* $\mathcal{D}$*. First, train the regression function on* $\mathcal{D}$*:* $\hat{f}_n \colon \mathcal{X} \to \mathcal{Y}$*. Then compute the residuals* $R_i = |Y_i - \hat{f}_n(X_i)|, i \in \mathcal{D}$*. For a new test point* (X_{n+1}, Y_{n+1})*, we have:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - \alpha \iff \mathbb{P}(R_{n+1} \leq \hat{q}_{n_2}) \geq 1 - \alpha \tag{D.15}$$

The last inequality only holds when R_{n+1} *is exchangeable with* $R_1, \ldots, R_n$*. This is* not *the case because the residuals* $R_i, i \in \mathcal{D}$ *will tend to be smaller since* $\hat{f}_n$ *was trained on* $\mathcal{D}$*, whereas* R_{n+1} *will be prone to be larger as* $\hat{f}_n$ *was not trained on* X_{n+1}*. Therefore,* $R_1, \ldots, R_n, R_{n+1}$ *are not exchangeable. Splitting the dataset into training and calibration allows for exchangeable residuals.*

Similarly to theorem 16, a training conditional coverage is possible for split conformal prediction, and was proven by Vovk (2012a).

Theorem 18 (Train-conditional coverage for split conformal prediction)
With the only assumption that $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ *are exchangeable and for any* $\delta \in (0, 1)$*, the the split conformal prediction set satisfies*[30]

$$\mathbb{P}[\text{COV}] \geq 1 - \delta \tag{D.16}$$

Where COV *is the event of satisfying the coverage guarantee:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1}) | \mathcal{D}_n\right) \geq 1 - \alpha \tag{D.17}$$

[30] The proof can be found in Vovk (2012a) see proposition 2a and in Bian et al. (2023) see theorem 1.

Split conformal prediction solves the main problem of full conformal prediction. It is less computationally demanding and achieves training-conditional coverage with minimal assumptions. Unfortunately, splitting the data is not desired at all, especially in our future use cases where we will not have a lot of data points. In between full and split conformal prediction we can leverage cross-validation techniques to (hopefully) keep the best of both worlds.

D.4 Cross-conformal prediction

Cross-conformal prediction was introduced by Vovk (2012b) and Vovk et al. (2019). It uses cross-validation instead of a single split. Let P be a distribution on $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(X_i, Y_i) \sim P$ and $(X_{n+1}, Y_{n+1}) \sim P$ be a test point. Let $K \in [\![2, n]\!]$ be the number of folds in the cross-validation, such that the full dataset $\mathcal{D}_n$ is divided into K datasets D_k, $k \in [\![1, K]\!]$ of equal size m.
For every fold, train a point prediction regression function on the remainder of the dataset:

$$\hat{f}_{-k} \colon \mathcal{X} \to \mathcal{Y}, \text{ trained on } \mathcal{D}_1, \ldots, \mathcal{D}_{k-1}, \mathcal{D}_{k+1}, \ldots, \mathcal{D}_K. \tag{D.18}$$

Next, compute the residuals on the hold-out fold $\mathcal{D}_k$:

$$\begin{aligned} R_{i,k} &= |Y_i - \hat{f}_{-k}(X_i)|, \ i \in \mathcal{D}_k \\ R^y_{n+1,k} &= |y - \hat{f}_{-k}(X_{n+1})| \end{aligned} \tag{D.19}$$

Here y is a trial value, similar to full conformal prediction.

Using those residuals, Vovk (2012b) proposed to compute the following prediction set:

$$\widehat{C}_n(X_{n+1}) = \left\{ y \colon \frac{1 + \sum_{k=1}^{K} \sum_{i \in \mathcal{D}_k} \mathbb{1}\left[R_{i,k} \leq R^y_{n+1,k}\right]}{n+1} > \alpha \right\} \tag{D.20}$$

First of all, this prediction set is not centered on any mean prediction (like are full and split methods) and is not necessary an interval. Second of all, there a *no* guarantee on the coverage nor marginal nor train-conditional.

Later, Vovk et al. (2019) presented a novel way of computing the prediction set $\widehat{C}_n$ changing the way the p-value is computed in D.20. They call their new model cross-conformal predictive systems. The new p-value is computed as follow:

$$p^y = \frac{\tau + \sum_{k=1}^{K} \sum_{i \in \mathcal{D}_k} \mathbb{1}\left[R_{i,k} < R^y_{n+1,k}\right] + \tau \mathbb{1}\left[R_{i,k} = R^y_{n+1,k}\right]}{n+1} \tag{D.21}$$

Then the prediction set becomes:

$$\widehat{C}_n(X_{n+1}) = \{ y \colon p^y > \alpha \} \tag{D.22}$$

This method is accompanied by the coverage theorem,

Theorem 19 (Marginal coverage for cross-conformal prediction (one p-value))
If $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ *are exchangeable, then for any* $K \in [\![2, n]\!]$ *the cross-conformal prediction set, constructed using D.22, satisfies*[31]*:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - 2\alpha - 2(1-\alpha)\frac{1 - \frac{1}{K}}{\frac{n}{K} + 1} \tag{D.23}$$

[31] A proof for this theorem can be found in Barber et al. (2020a) in section B.1.2.

The proof relies on a slightly modified cross-conformal prediction set, that we present here for completeness. Compute the p-values for each folds individually:

$$p^y_k = \frac{\tau + \sum_{i \in \mathcal{D}_k} \mathbb{1}\left[R_{i,k} < R^y_{n+1,k}\right] + \tau \mathbb{1}\left[R_{i,k} = R^y_{n+1,k}\right]}{m+1} \tag{D.24}$$

The new prediction set is:

$$\widehat{C}_n(X_{n+1}) = \left\{ y \colon \frac{1}{K} \sum_{k=1}^{K} p^y_k > \alpha \right\} \tag{D.25}$$

Theorem 20 (Marginal coverage for cross-conformal prediction (mean of p-values))
If $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ *are exchangeable, then for any* $K \in [\![2, n]\!]$ *the cross-conformal prediction set, constructed using D.25, satisfies*[32]*:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - 2\alpha \tag{D.26}$$

[32] A proof for this theorem can be found in Barber et al. (2020a) in section B.1.2.

This completes the overview of cross-conformal prediction, which are not as popular as other methods. To the best of my knowledge, there are no train-conditional results for such cross-conformal prediction.

D.5 Jackknife

Jackknife is the name given to the leave-one-out cross-validation. In the conformal world, the jackknife model is different from the cross-conformal prediction with $K = n$. Let $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(X_i, Y_i) \sim P$ where P is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(X_{n+1}, Y_{n+1}) \sim P$ be a test point. First of all, train a point predictor regression function on the whole training dataset $\mathcal{D}_n$:

$$\hat{f}_n \colon \mathcal{X} \to \mathcal{Y} \tag{D.27}$$

Then, for every observation, train the following hold-out train predictor regression function

$$\hat{f}_{-i} \colon \mathcal{X} \to \mathcal{Y}, \quad \text{trained on } \mathcal{D}_n \backslash \{(X_i, Y_i)\} \tag{D.28}$$

and compute the residuals

$$R_i = |Y_i - \hat{f}_{-i}(X_i)|, \ i \in [\![1, n]\!] \tag{D.29}$$

Then, apply the finite sample correction to the R_i's to compute the quantile:

$$\hat{q}_n = Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n}\sum_{i=1}^{n} \delta_{R_i}\right) \tag{D.30}$$

Finally, the prediction set writes[33]:

[33]

$$\widehat{C}_n(X_{n+1}) = \left[\hat{f}_n(X_{n+1}) - \hat{q}_n, \ \hat{f}_n(X_{n+1}) + \hat{q}_n\right] \tag{D.31}$$

Unfortunately for this version of jackknife conformal prediction there are no theoretical results on coverage (marginal or train-conditional), see Lei et al. (2017). To go further, Steinberger et al. (2016), Steinberger et al. (2022), and Amann et al. (2023) discusses coverage guarantee for the jackknife method, under stricter assumptions.

D.6 Jackknife+

Barber et al. (2020a) introduced a new idea to the jackknife method, which allows for coverage guarantees. They extend this idea to the K-fold cross-validation model and present a new one called minmax. Let $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(X_i, Y_i) \sim P$ where P is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(X_{n+1}, Y_{n+1}) \sim P$ be a test point.
Jackknife+ is very similar to jackknife except that we do not need to compute the point prediction regression function trained on the whole $\mathcal{D}_n$ dataset. Simply train the leave-one-out point predictors:

$$\hat{f}_{-i} \colon \mathcal{X} \to \mathcal{Y}, \quad \text{trained on } \mathcal{D}_n \backslash \{(X_i, Y_i)\} \tag{D.32}$$

and compute the leave-one-out residuals

$$R_i = |Y_i - \hat{f}_{-i}(X_i)|, \ i \in [\![1, n]\!] \tag{D.33}$$

The trick in Jackknife+ is to compute not one but two quantiles and to use the leave-one-out point predictors instead of $\hat{f}_n$. For the test point X_{n+1}, compute the following:

$$\begin{aligned} E_i^- &\coloneqq \hat{f}_{-i}(X_{n+1}) - R_i, \ i \in [\![1, n]\!] \text{ and} \\ E_i^+ &\coloneqq \hat{f}_{-i}(X_{n+1}) + R_i, \ i \in [\![1, n]\!]. \end{aligned} \tag{D.34}$$

Then, for the lower bound of the prediction set compute the α adjusted quantile of E_i^-

$$\hat{q}_n^- = Q\left(\frac{\lceil \alpha(n+1) \rceil}{n}, \frac{1}{n}\sum_{i=1}^{n} \delta_{E_i^-}\right), \tag{D.35}$$

and for the upper bound compute the $1-\alpha$ adjusted quantile of E_i^+

$$\hat{q}_n^+ = Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n}\sum_{i=1}^{n} \delta_{E_i^+}\right). \tag{D.36}$$

The prediction set then writes:

$$\widehat{C}_n(X_{n+1}) = \left[\hat{q}_n^-,\, \hat{q}_n^+\right] \tag{D.37}$$

Theorem 21 (Marginal coverage for jackknife+)
If $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ are exchangeable, then the jackknife+ prediction set satisfies[34]*:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - 2\alpha \tag{D.38}$$

[34] A proof for this theorem is to be found in Barber et al. (2020a) in section 6.

Barber et al. (2020a) provides another theorem on the jackknife+ method that states that the bound $1-2\alpha$ is the best the jackknife+ method can achieve. To achieve better bounds one must use additional assumptions. In the same line, Bian et al. (2023) shows that without further assumptions the jackknife+ can not achieve train-conditional coverage. Finally, R. Liang et al. (2024) show training-conditional coverage for the jackknife+ using an algorithm stability assumption. Writing $\beta_{m,n}^{\text{out}}$ the out-of-sample m-stability, as it is defined in R. Liang et al. (2024) (see definition 3.1). Their theorem writes:

Theorem 22 (Train-conditional coverage for jackknife+)
Suppose the algorithm used to train the point predictor regression functions is symmetric and assume that $Y|X$ has density $f_{Y|X}(y|x)$ with respect to the Lebesgue measure and define

$$B_{dens} = \mathbb{E}\left[\sup_{y \in \mathbb{R}} f_{Y|X}(y|X)\right] \tag{D.39}$$

Then, for any $m \geq 1$ and $\delta \in (0,1)$[35]

$$\mathbb{P}\left[\text{COV}\right] \geq 1 - 3\delta - \sqrt[4]{2B_{dens}\beta_{m,n-1}^{out}}. \tag{D.40}$$

[35] Proof is given in R. Liang et al. (2024).

Where COV *is the event of the training-conditional coverage guarantee,*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n^{\gamma}(X_{n+1}) | \mathcal{D}_n\right) \geq 1 - \alpha - 3\sqrt{\frac{\log(1/\delta)}{2\min(n,m)}} - 4\sqrt[4]{2B_{dens}\beta_{m,n-1}^{out}} \tag{D.41}$$

D.6.1 Minmax variant

In order to address the weaker coverage guarantee of theorem 3, Barber et al. (2020a) suggested another method, jackknife minmax.
Simply compute the adjusted quantile of the leave-one-out residuals

$$\hat{q}_n = Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n}\sum_{i=1}^{n} \delta_{R_i}\right), \tag{D.42}$$

and compute the prediction set

$$\widehat{C}_n(X_{n+1}) = \left[\min_{i=1,\ldots,n} \hat{f}_{-i}(X_{n+1}) - \hat{q}_n,\, \max_{i=1,\ldots,n} \hat{f}_{-i}(X_{n+1}) + \hat{q}_n\right] \tag{D.43}$$

The jackknife minmax is much more conservative than jackknife+ and satisfies the following theorem:

Theorem 23 (Marginal coverage for jackknife-minmax)
If $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ are exchangeable, then the jackknife-minmax prediction set satisfies[36]*:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - \alpha \tag{D.44}$$

[36] A proof for this theorem is to be found in Barber et al. (2020a) in section 2.2.

D.6.2 K-fold CV+ variant

Barber et al. (2020a) also adapted their method to cross-conformal prediction. Using the same notations from section D.4,

$$\begin{aligned} &\hat{f}_{-k} \colon \mathcal{X} \to \mathcal{Y}, \text{ trained on } \mathcal{D}_1, \ldots, \mathcal{D}_{k-1}, \mathcal{D}_{k+1}, \ldots, \mathcal{D}_K. \\ &R_{i,k} = |Y_i - \hat{f}_{-k}(X_i)|, \ i \in \mathcal{D}_k \end{aligned} \tag{D.45}$$

Similarly to jackknife+ define:

$$\begin{aligned} E^-_{i,k} &\coloneqq \hat{f}_{-k}(X_{n+1}) - R_{i,k}, \ i \in \mathcal{D}_k, \ k \in [\![1, K]\!] \text{ and} \\ E^+_{i,k} &\coloneqq \hat{f}_{-k}(X_{n+1}) + R_{i,k}, \ i \in \mathcal{D}_k, \ k \in [\![1, K]\!]. \end{aligned} \tag{D.46}$$

Then, compute the lower and upper quantiles:

$$\begin{aligned} \hat{q}^-_n &= Q\left(\frac{\lfloor \alpha(n+1) \rfloor}{n}, \frac{1}{n}\sum_{k=1}^{K}\sum_{i \in \mathcal{D}_k} \delta_{E^-_{i,k}}\right), \\ \hat{q}^+_n &= Q\left(\frac{\lceil (1-\alpha)(n+1) \rceil}{n}, \frac{1}{n}\sum_{k=1}^{K}\sum_{i \in \mathcal{D}_k} \delta_{E^+_{i,k}}\right). \end{aligned} \tag{D.47}$$

Finally the prediction set is computed as

$$\widehat{C}_n(X_{n+1}) = \left[\hat{q}^-_n, \ \hat{q}^+_n\right], \tag{D.48}$$

and satisfies the theorem:

Theorem 24 (Marginal coverage for K-fold CV+)
If $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$ are exchangeable, then the K-fold CV+ prediction set satisfies[37]*:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - 2\alpha - \min\left\{\frac{2(1-1/K)}{n/K+1}; \frac{1-K/n}{K+1}\right\}. \tag{D.49}$$

The K-fold CV+ method also has a few assumption based training-conditional coverage, presented in Bian et al. (2023).

[37] A proof for this theorem is to be found in Barber et al. (2020a) see theorem 4.

Theorem 25
For any $K \geq 2$, for any $m \geq 1$ such that every fold has m elements. For any $\delta > 0$ the K-fold CV+ prediction set satisfies[38]*:*

$$\mathbb{P}\left[\text{COV}\right] \geq 1 - \delta \tag{D.50}$$

Where COV *is the event of satisfying the coverage guarantee:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}^{\gamma}_n(X_{n+1}) | \mathcal{D}_n\right) \geq 1 - 2\alpha - \sqrt{\frac{2\log(K/\delta)}{m}} \tag{D.51}$$

[38] The proof is provided in Bian et al. (2023).

Remark 9. *Using jackknife and cross-validation can seem computationally inefficient but some machine learning models do not need to be refitted to compute the leave-one-out residuals (linear smoother) and for random forest, the leave-one-out predictors can be computed at low cost.*

With that we cover most of the results in conformal prediction. In the next two section we are going to discuss first, adaptativity of conformal prediction methods and second, other research trends or concepts that we mention here for future reference.

D.7 Discussion on adaptativity

Adaptativity of the prediction bands was defined as key property for prediction set regression functions. Unfortunately, for all methods discussed above, the prediction set is of constant width[39]. Adaptativity of the prediction set can be defined as the ability to have a width that is a function of x. The question is, is it possible to have adaptative prediction sets and is there a simple way to achieve that in conformal prediction methods ?

[39] Except for full conformal prediction, but in practice the prediction set is almost of constant width.

The answer is yes. For all conformal methods (full, split, cross, jackknife, jackknife+ and variants) we presented them using residuals (absolute value between truth and prediction) to measure how good or bad the point predictor was. This does not need to be the case and we can replace the residuals with a more general score metric of our choice. Most of the methods we are going to present next are "scaled" version of the residuals, and the difference between them lies in how the scale factor is computed, with the exception of conformalized quantile regression that we are going to present last. In the following, we will use the letter S for the scores to replace the R of the residuals.

D.7.1 Studentized residuals

Simply replace the computation of residuals by the following score:

$$S_i = \frac{|Y_i - f(\hat{X}_i)|}{\hat{\sigma}(X_i)},\ i \in \mathcal{D}_{\text{cal}}. \tag{D.52}$$

Where $\mathcal{D}_{\text{cal}}$ is the calibration dataset used in each specific conformal method, and $\hat{\sigma}$ is an estimation of the standard deviation[40] of $|Y - f(\hat{X})|$. We can then compute the finite sample correction on the scores $S_i,\ i \in \mathcal{D}_{\text{cal}}$:

[40] Lei et al. (2017) proposes to compute $\hat{\sigma}$ as the conditional mean absolute deviation of $\left(Y - f(\hat{X})\right) | X = x$.

$$\hat{q}_{\text{cal}} = Q\left(\frac{\lceil (1-\alpha)(n_{\text{cal}}+1)\rceil}{n_{\text{cal}}}, \frac{1}{n_{\text{cal}}} \sum_{i \in n_{\text{cal}}} \delta_{S_i}\right) \tag{D.53}$$

The prediction set is constructed the exact same way:

$$\widehat{C}_n(X_{n+1}) = \left\{y:\ S^y_{n+1} \leq \hat{q}_{\text{cal}}\right\} \tag{D.54}$$

Which for split conformal prediction can be rewritten:

$$\widehat{C}_n(X_{n+1}) = \left[\hat{f}(X_{n+1}) \pm \hat{\sigma}(X_{n+1}) \cdot \hat{q}_{\text{cal}}\right] \tag{D.55}$$

D.7.2 Bagging methods

Without going over a full course on bagging models in machine learning (Breiman (1996) and Breiman (2001)), the idea is to aggregate a number of weak learners that were trained concurrently.

In regression the aggregation that leads to the final estimator is (usually) the average:

$$\hat{f}_B(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(x) \tag{D.56}$$

where $\hat{f}_b$, $b = 1, \dots B$ are the weak learners and B is the number of weak learners we want to train. For conformal methods, one can use the estimated bagging variance:

$$\hat{\sigma}_B(x) = \sqrt{\frac{1}{B-1} \sum_{b=1}^{B} \left(\hat{f}_b(x) - \hat{f}_B(x)\right)^2} \tag{D.57}$$

The conformal scores become:

$$S_i = \frac{|Y_i - f(\hat{X}_i)|}{\hat{\sigma}_B(X_i)},\ i \in \mathcal{D}_{\text{cal}}. \tag{D.58}$$

The prediction set is then computed following the conformal variant. For references see Johansson et al. (2014)

D.7.3 Conformalized Gaussian Processes

As discussed in 2.5, Gaussian processes are widely used in machine learning and especially meta-modelling. One of the main differences between Gaussian processes and other machine learning models is that the (point) prediction at a test point X_{n+1} is given by the mean of a normal distribution with mean:

$$\hat{f}(X_{n+1}) = \mu(X_{n+1}) + k(X, X_{n+1})k(X, X)^{-1}(y - \mu(X)) \tag{D.59}$$

This normal distribution also comes with the following variance:

$$\hat{\sigma}^2(X_{n+1}) = k(X_{n+1}, X_{n+1}) - k(X, X_{n+1})k(X, X)k(X, X_{n+1}) \tag{D.60}$$

The idea is then to normalise the score by the posterior variance of the Gaussian process:

$$S_i = \frac{|Y_i - f(\hat{X}_i)|}{\max(\hat{\sigma}(X_i), \varepsilon)},\ i \in \mathcal{D}_{\text{cal}} \text{ and } \varepsilon > 0. \tag{D.61}$$

The value ε is here to prevent division by 0 when the Gaussian process interpolates. In Rasmussen et al. (2006) can be found formulas for leave-one-out Gaussian process regression. Both Harris Papadopoulos (2023) and Jaber et al. (2024) take advantage of those to propose a complete conformalized version of the Gaussian Process Regression model using full, split and jackknife+ conformal variants. Finally, Pion et al. (2024), introduced asymmetric scores[41] for the conformalized Gaussian process model, building upon Jaber et al. (2024) work.

[41] The score proposed is D.61 without the absolute values.

D.7.4 Conformalized Quantile Regression

The idea of Conformalized Quantile Regression (Romano et al. (2019)) is slightly different from what we have discussed so far. Instead of estimating the mean function of the data, Conformalized Quantile Regression leverages quantile regression models, discussed in 2.4, to compute the conformal scores.

For the split conformal quantile regression the method goes as follow. Let $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ an exchangeable dataset such that for all $i \in [\![1, n]\!]$, $(X_i, Y_i) \sim P$ where P is a distribution on $\mathcal{X} \times \mathcal{Y}$ and $(X_{n+1}, Y_{n+1}) \sim P$ be a test point. Divide the training dataset into two, a proper training dataset $\mathcal{D}_{n_1}$ and $\mathcal{D}_{n_2}$ a calibration dataset. Using an algorithm of choice for quantile regression, estimate the quantile regression function at two quantiles, α_{lo} and α_{hi} such that $\alpha_{\text{lo}} + \alpha_{\text{hi}} = \alpha$, the targeted error rate.

$\hat{q}_{\alpha_{\text{lo}}} : \mathcal{X} \to \mathcal{Y}$, estimation of the α_{lo} quantile function trained on $\mathcal{D}_{n_1}$.
$\hat{q}_{\alpha_{\text{hi}}} : \mathcal{X} \to \mathcal{Y}$, estimation of the α_{hi} quantile function trained on $\mathcal{D}_{n_1}$.

The scores are then computed in the following manner:

$$S_i = \max\left\{\hat{q}_{\alpha_{\text{lo}}}(X_i) - Y_i;\ Y_i - \hat{q}_{\alpha_{\text{hi}}}(X_i)\right\},\ i \in \mathcal{D}_{n_2}. \tag{D.62}$$

Next, compute the the finite sample correction quantile on the calibration dataset: $\hat{q}_{n_2}$. And the prediction set writes:

$$\widehat{C}_n(X_{n+1}) = [\hat{q}_{\alpha_{\text{lo}}}(X_{n+1}) - \hat{q}_{n_2};\ \hat{q}_{\alpha_{\text{hi}}}(X_{n+1}) + \hat{q}_{n_2}] \tag{D.63}$$

Romano et al. (2019) proves the following theorem:

Theorem 26 (Marginal coverage for Conformalized Quantile Regression) *If* $(X_i, Y_i), i = 1 \ldots, n, n+1$ *are exchangeable, the prediction interval constructed using split CQR satisfies:*

$$\mathbb{P}\left(Y_{n+1} \in \widehat{C}_n(X_{n+1})\right) \geq 1 - \alpha. \tag{D.64}$$

What is not stated in the original paper and would be interesting to investigate is, i) can CQR be adapted to the jackknife+ algorithm, ii) can CQR achieve train-conditional coverage and under which assumptions ?

Appendix E

More on the simultaneous model

Solving for the disturbance

As we can see in figures 6.3 and 6.5, the mean estimation is completely off. Even thought the Energy-HSIC chooses a length scale that has both great mean estimation and great prediction bands estimation, for high length scales the model is off. To solve this defect we came up with an idea, to constraint the mean prediction even more. In problems 6.3a and 6.4a we already used a pre-fitted Gaussian process to estimate the hyper-parameters s and the mean length scale. We decided to constraint the quadratic errors of our mean estimation by the quadratic errors of the Gaussian process[42] on the sample used for training the semi-definite programming problem. Writing the problems down, we have:

[42] The same one we already use to define the hyper-parameters values for the length scale and the norm of the mean function.

Liang:

$$\min_{\gamma\in\mathbb{R}^n, A\in\mathbb{S}^n} \quad \operatorname{Tr}(A) + \lambda_2 \operatorname{Tr}(A^2) \tag{E.1a}$$

$$\text{s.t. } \tilde{v}_A(x_i) \geq \left(y_i - \langle K^m_{x_i}, \gamma\rangle\right)^2,\ i \in [\![1, n]\!] \tag{E.1b}$$

$$\gamma^\top K^m \gamma \leq s \tag{E.1c}$$

$$\left(y_i - \langle K^m_{x_i}, \gamma\rangle\right)^2 \leq c_i,\ i \in [\![1, n]\!] \tag{E.1d}$$

$$A \succeq 0 \tag{E.1e}$$

Fan:

$$\min_{\gamma\in\mathbb{R}^n, A\in\mathbb{S}^n} \quad \operatorname{Tr}(K^v A) + \lambda_2 \operatorname{Tr}(A^2) \tag{E.2a}$$

$$\text{s.t. } \tilde{v}_A(x_i) \geq \left(y_i - \langle K^m_{x_i}, \gamma\rangle\right)^2,\ i \in [\![1, n]\!] \tag{E.2b}$$

$$\operatorname{Tr}(A) \leq r \tag{E.2c}$$

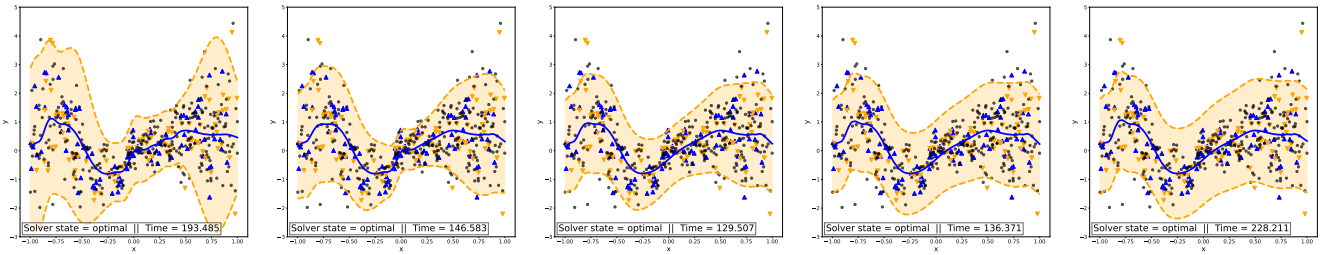
$$\gamma^\top K^m \gamma \leq s \tag{E.2d}$$

$$\left(y_i - \langle K^m_{x_i}, \gamma\rangle\right)^2 \leq c_i,\ i \in [\![1, n]\!] \tag{E.2e}$$

$$A \succeq 0 \tag{E.2f}$$

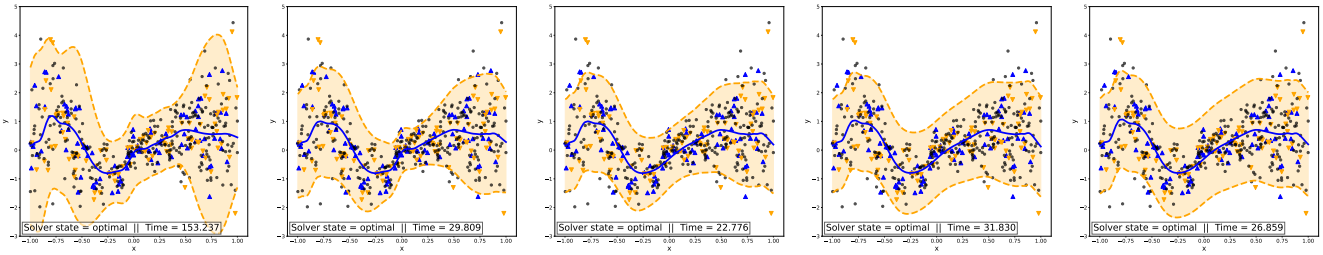
Where $c_i = \left(y_i - \widehat{m}_{\mathrm{GP}}(x_i)\right)^2$, if we denote by $\widehat{m}_{\mathrm{GP}}$ the Gaussian process model.

Because we only have one problem where the mean function is not null, we are going to focus ourselves on the experiment 4. Figure E.1 shows Liang's problem E.1a and figure E.2 showcases Fan's problem E.2a. As we can see, the estimated mean function is now really close to the one we



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure E.1: Experiment 4 - Liang, simultaneous estimation with quadratic constraints - $\lambda_2 = 1; \delta = 0.001$ - Evolution of the estimated mean function and prediction bands with the length scale.



(a) length scale = 0.2 **(b)** length scale = 0.4 **(c)** length scale = 0.6 **(d)** length scale = 0.8 **(e)** length scale = 1.0

Figure E.2: Experiment 4 - Fan, simultaneous estimation with quadratic constraints - $\lambda_2 = 1; \delta = 0.001$ - Evolution of the estimated mean function and prediction bands with the length scale.

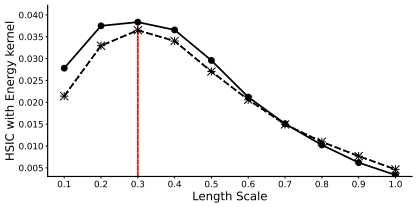


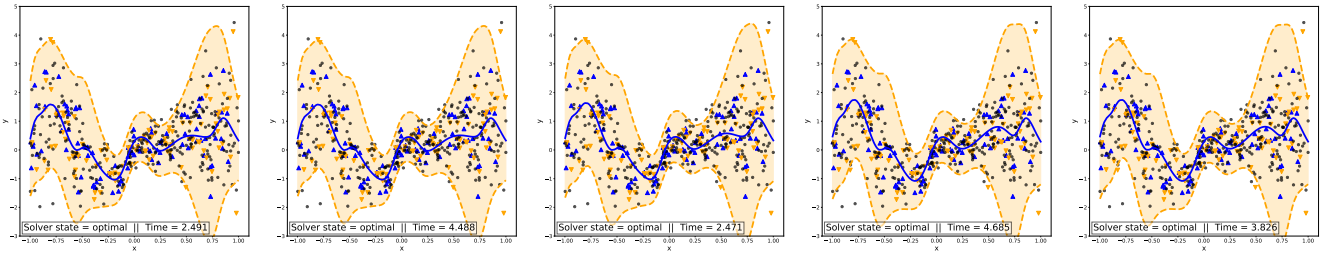
Figure E.3: Energy-HSIC between residuals and prediction bands's width (on calibration data) in figure E.1 and E.2, meaning, with quadratic constraints.

had with the "sequential" procedure and is consistent across multiple value of length scale for the prediction bands RKHS. And looking at figure E.3, the Energy-HSIC still chooses adaptative prediction bands.

Impact of the regularisation on the bands' shape

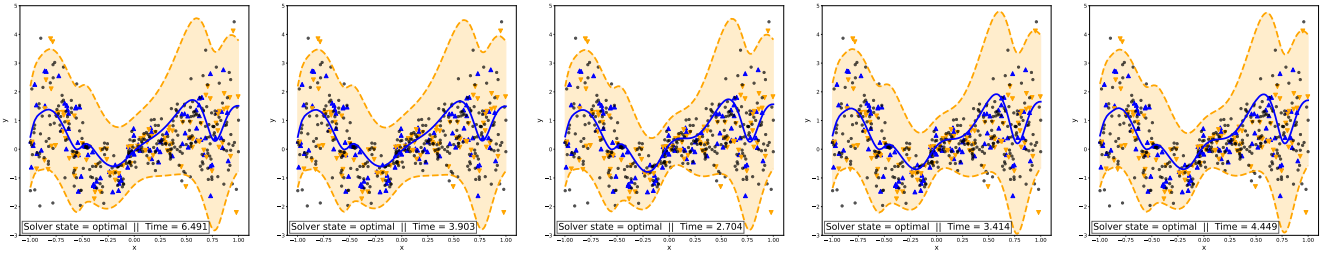
In this section we will explore the impact of the regularisation coefficient λ_2 on the prediction bands' shape. We chose five different values: $0, 1e^{-6}, 1, 1e^{3}, 1e^{6}$. For this we are going to study experiment four only. Figure E.4 we showcase Liang's objective function with length scale 0.2 and figure E.5 for length scale 0.4. Finally figure E.6 presents Fan's objective function with length scale 0.2.

What we can say about those figures is that the overall shape of the bands do not drastically change will the value of λ_2. But figure E.6 shows that *it can* modify the bands' shape. However, we just saw that most of our metrics were sensible to slight variations in the prediction bands. Thus, λ_2 can have an impact on the performance of the metric to tune the length scale of the variance kernel.



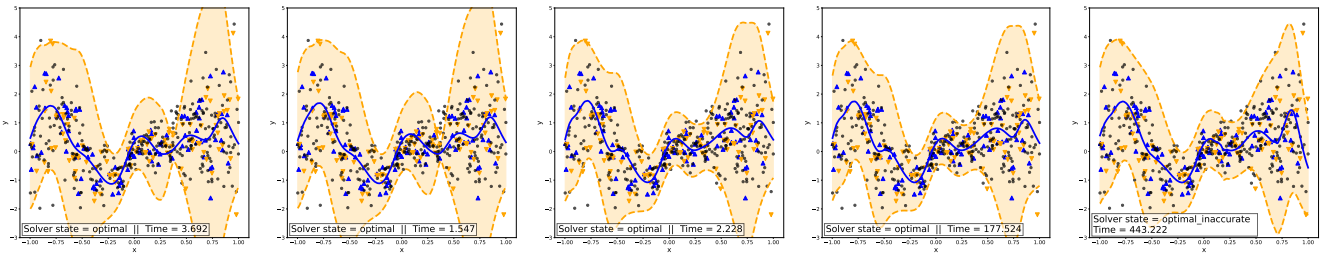
(a) $\lambda_2 = 0$ **(b)** $\lambda_2 = 1e^{-6}$ **(c)** $\lambda_2 = 1$ **(d)** $\lambda_2 = 1000$ **(e)** $\lambda_2 = 1e^{6}$

Figure E.4: Experiment 4 - Liang, simultaneous - $ls = 0.2; \delta = 0.001$ - Evolution of the estimated mean function and prediction bands with the length scale.



(a) $\lambda_2 = 0$ **(b)** $\lambda_2 = 1e^{-6}$ **(c)** $\lambda_2 = 1$ **(d)** $\lambda_2 = 1000$ **(e)** $\lambda_2 = 1e^{6}$

Figure E.5: Experiment 4 - Liang, simultaneous - $ls = 0.4; \delta = 0.001$ - Evolution of the estimated mean function and prediction bands with the length scale.



(a) $\lambda_2 = 0$ **(b)** $\lambda_2 = 1e^{-6}$ **(c)** $\lambda_2 = 1$ **(d)** $\lambda_2 = 1000$ **(e)** $\lambda_2 = 1e^{6}$

Figure E.6: Experiment 4 - Fan, simultaneous - $ls = 0.2; \delta = 0.001$ - Evolution of the estimated mean function and prediction bands with the length scale.