UPPSALA
UNIVERSITET

IT Licentiate theses
2019-005

Tailoring Gaussian processes for tomographic reconstruction

CARL JIDLING

UPPSALA UNIVERSITY
Department of Information Technology

UPPSALA
UNIVERSITET

Tailoring Gaussian processes for tomographic reconstruction

Carl Jidling
carl.jidling@it.uu.se

October 2019

Division of Systems and Control
Department of Information Technology
Uppsala University
Box 337
SE-751 05 Uppsala
Sweden

http://www.it.uu.se/

Dissertation for the degree of Licentiate of Philosophy in Electrical Engineering with specialisation in Signal Processing

ISSN 1404-5117
Printed by the Department of Information Technology, Uppsala University, Sweden

Abstract

A probabilistic model reasons about physical quantities as random variables that can be estimated from measured data. The Gaussian process is a respected member of this family, being a flexible non-parametric method that has proven strong capabilities in modelling a wide range of nonlinear functions. This thesis focuses on advanced Gaussian process techniques; the contribution consist of practical methodologies primarily intended for inverse tomographic applications.

In our most theoretical formulation, we propose a constructive procedure for building a customised covariance function given any set of linear constraints. These are explicitly incorporated in the prior distribution and thereby guaranteed to be fulfilled by the prediction.

One such construction is employed for strain field reconstruction, to which end we successfully introduce the Gaussian process framework. A particularly well-suited spectral based approximation method is used to obtain a significant reduction of the computational load. The formulation has seen several subsequent extensions, represented in this thesis by a generalisation that includes boundary information and uses variational inference to overcome the challenge provided by a nonlinear measurement model.

We also consider X-ray computed tomography, a field of high importance primarily due to its central role in medical treatments. We use the Gaussian process to provide an alternative interpretation of traditional algorithms and demonstrate promising experimental results. Moreover, we turn our focus to *deep kernel learning*, a special construction in which the expressiveness of a standard covariance function is increased through a neural network input transformation. We develop a method that makes this approach computationally feasible for integral measurements, and the results indicate a high potential for computed tomography problems.

Acknowledgments

This thesis is a result of united efforts, and I thank all of you who made it happen: my main supervisor Thomas Schön, for your enthusiastic support and guidance; my co-supervisors Niklas Wahlström and Adrian Wills, for your commitment and dedication; and my remaining co-authors for the collaborations and joint work on which the thesis resides.

This research is financially supported by the Swedish Foundation for Strategic Research (SSF) via the project *ASSEMBLE* (Contract number: RIT 15-0012). The work is also supported by the Swedish Research Council (VR) via the projects *Probabilistic modelling of dynamical systems* (Contract number: 621-2013-5524) and *Learning flexible models for nonlinear dynamics* (contract number: 2017-03807).

List of Papers

This thesis includes the following papers:

I C. Jidling, N. Wahlström, A. Wills and T. B. Schön. “Linearly constrained Gaussian processes”. In: *Advances in Neural Information Processing Systems (NIPS).* Los Angeles, US, 2017, pp. 1215–1224.

II C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. “Probabilistic modelling and reconstruction of strain”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

III Z. Purisha, C. Jidling, N. Wahlström, T. Schön and S. Särkkä. “Probabilistic approach to limited-data computed tomography reconstruction”. In: *Inverse Problems* 35.10 (2019), p. 105004.

IV C. Jidling, T. B. Schön, J. Hendriks and A. Wills. *Deep kernel learning for integral measurements.* Tech. rep. arXiv:1909.01844, Sept. 2019.

V J. N. Hendriks, C. Jidling, T. B. Schön, A. Wills, C. M. Wensrich and E. H. Kisi. “Neutron transmission strain tomography for non-constant stress-free lattice spacing”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 456 (2019), pp. 64–73.

Contents

Chapter 1

Introduction

1.1 Background and general motivation

Throughout history, humans have carried a profound need and desire to understand the world we live in and the universe in which it resides. Science is the guiding star in this process, characterised by systematic reproducible research, encouraging questioning that favours new insights and understandings.

Due to the studies and observations of natural phenomena, a bit of effort has been put into the development of theoretical models that generalise their behaviour. We have learnt to make practical use of these models; the result so far is stunning and overwhelmingly demonstrated by the high-technological society surrounding us.

The focus of the thesis is the specific mathematical model known as the *Gaussian process*. It belongs to the class of *Bayesian non-parametric* methods, which employ a probabilistic viewpoint and treat physical quantities as random variables – intuitively abstract, but again practically useful. Although the Gaussian process was originally developed decades ago, its position has been strengthened by the increasingly massive interest in machine learning.

The traditional way of describing a physical system is by relating the involved quantities through a customised mathematical formulation – not more complicated than necessary – that makes use of knowledge gained from experience and empirical observations. Force equals mass times acceleration, voltage equals current times resistance. And so on. This is the so called *model-driven* approach.

Within the machine learning community, a rather opposite idea is exploited. Here the cornerstone is advanced mathematical models with general descriptive capabilities. These flexible models are being fed with measured data, from which they *learn* the underlying relations. Consequently, we refer to these methods as *data-driven*.

The Gaussian process is a member of this latter class; it is practically useless in the absence of data, which partially defines the model itself. However, it still turns out that it is possible to embed essential physical information inside this construction, and so improve its modelling performance. The intersection between model and data-driven approaches is exciting, and has a potential of which we may yet only have scratched the surface. This thesis does some of that scratching. I hope you enjoy it.

1.2 Contributions

The main contributions of this thesis are as follows.

- Paper I: a constructive procedure for explicit inclusion of linear constraints in the Gaussian process.
- Paper II: a tailored Gaussian process model customised for strain field modelling.
- Paper III: a thorough analysis of the Gaussian process as a tool within X-ray computed tomography.
- Paper IV: a practical approach that enables deep kernel learning for integral measurements.
- Paper V: a novel methodology that generalises strain field modelling by accounting for nonlinear effects.

1.3 Outline of the thesis

The first part of the thesis contains background theory and introduces the relevant problem formulations. Chapter 2 focuses on the very basics of Gaussian process modelling, while more advanced extensions are considered in Chapter 3. These are necessary when attacking the tomographic problems described in Chapter 4. We conclude the first part in Chapter 5 and briefly discuss topics of future work.

The remaining second part is constituted by five papers, which are summarised below.

Paper I: Linearly constrained Gaussian processes

> C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. "Probabilistic modelling and reconstruction of strain". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

Summary: This paper describes a practical method for embedding linear constraints in a Gaussian process, which *explicitly ensures* that they are fulfilled by the resulting prediction. The construction relies on the existence of an underlying potential function that is mapped to the true function through a linear operator transformation, which can be found in a straightforward manner. We compare the performance to the naive and numerically problematic approach of including the constraints as pointwise, fictitious measurements. The practical potential is demonstrated for the problem of magnetic field modelling.

Statement of contribution: The development of the approach presented in this paper involved all authors. Carl Jidling and Niklas Wahlström was equally responsible for the implementations and most of the writing, with important contributions and feedback provided by Adrian Wills and Thomas B. Schön.

Paper II: Probabilistic modelling and reconstruction of strain

> C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. "Probabilistic modelling and reconstruction of strain". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

Summary: In this paper we use a linearly constrained Gaussian process for reconstructing strain fields in deformed materials. This is an inverse tomographic problem with data generated from neutron-based *Bragg-edge* experiments, mathematically described as line integrals of the projected strain tensor. By designing our model with respect to the underlying *Airy stress function*, we explicitly incorporate the essential *equilibrium constraints*. We exploit an approximative framework that not only allows us to work with large data sets, but also provides closed form expressions of what would otherwise have been expensive numerical double integral evaluations. This work breaks new ground and serves as a starting point of probabilistic strain field modelling, and has inspired several subsequent developments.

Statement of contribution: The theoretical construction described in this paper emerged from discussions involving all authors with contributions well reflected in the author ordering. The implementation and writing was done by Carl Jidling mainly, although with important contributions from the remaining authors, especially regarding the physics descriptions and related work in that field.

Paper III: Probabilistic approach to limited-data computed tomography reconstruction

> Z. Purisha, C. Jidling, N. Wahlström, T. Schön and S. Särkkä. "Probabilistic approach to limited-data computed tomography reconstruction". In: *Inverse Problems* 35.10 (2019), p. 105004.

Summary: This paper gives a thorough analysis of the Gaussian process in solving the inverse X-ray computed tomography problem. A spectral-based basis function expansion provides an alternative interpretation of classical methods such as Tikhonov regularisation, and we compare several systematic data-driven procedures for the hyperparameter tuning. Focusing on the *limited data setting*, we demonstrate an improved image quality as compared to the very commonly used *filtered back projection* algorithm.

Statement of contribution: The method proposed in this paper was developed by the authors jointly. The main part of the work was done by Zenith Purisha, including the production and compilation of experimental results, as well as the majority of the writing. Carl Jidling contributed to the implementation of the GP routines, assisted with parts of the writing and provided feedback during the revision process.

Paper IV: Deep kernel learning for integral measurements

> C. Jidling, T. B. Schön, J. Hendriks and A. Wills. *Deep kernel learning for integral measurements.* Tech. rep. arXiv:1909.01844, Sept. 2019.

Summary: Standard stationary covariance functions in Gaussian process modelling have a limited expressiveness; *deep kernel learning* increases it by first transforming the inputs through a neural network. Although simple in theory, the approach is practically challenging in problems when the data consist of integral measurements of the unknown function. We propose a method that allows for feasible implementations in these cases, including a customised hyperparameter initialisation obtained by pre-training the neural network. The performance is illustrated on X-ray computed tomography problems.

Statement of contribution: The idea to this paper is credited Johannes

Hendriks. Carl Jidling undertook most of the implementation and the majority of the writing, with essential contributions from Hendriks. Fruitful comments and feedback were provided by the remaining authors.

Paper V: Neutron Transmission Strain Tomography for Non-Constant Stress-Free Lattice Spacing

> J. N. Hendriks, C. Jidling, T. B. Schön, A. Wills, C. M. Wensrich and E. H. Kisi. "Neutron transmission strain tomography for non-constant stress-free lattice spacing". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 456 (2019), pp. 64–73.

Summary: Previous work on Gaussian processes for strain field reconstruction assume a constant lattice spacing between the crystal planes in the undeformed material. Although this is valid in many cases, it also limits the practical applications. However, the generalisation is challenging since it contains a nonlinear measurement model. In this paper we propose a method that models the lattice spacing and the strain field as a joint Gaussian process, and employ a variational approach in the inference step. The prediction is guaranteed to fulfil the equilibrium constraints, and convergence is ensured through the inclusion of boundary conditions. Our experimental results show a superior performance in comparison to the naive approach of incorrectly assuming a constant lattice spacing.

Statement of contribution: The conceptual method to solve this problem was developed by Johannes Hendriks, Carl Jidling, Thomas Schön, and Adrian Wills, with the implementation undertaken by Johannes Hendriks. The technical background dealing with crystal structure and Bragg-edge neutron transmission techniques was provided by Chris Wensrich, Erich Kisi, and Johannes Hendriks. Overall, contributions to the work are well reflected in the author ordering. Carl Jidling provided essential background in nonlinear methods for Gaussian processes and techniques for hyper parameter optimisation.

Chapter 2

Gaussian processes

The Gaussian process (GP) [27] has a prominent position within the machine learning community. Although it relies upon the somewhat abstract idea of modelling a function as a probabilistic object, it provides tractable mathematical expressions. While our interest lies in a continuous function, in practise we are only interested in predicting its values at a finite set of input locations. By modelling these function values as Gaussian random variables, we can construct their joint Gaussian distribution, no matter if they have been observed or not – and the desired quantities are obtained in closed form.

The GP is mainly used for regression and classification, although the latter field is today dominated by deep learning techniques. Indeed, the interest in this thesis is exclusively the regression problem. The GP has gained in popularity due to several attractive properties, including:

i) **Encoding a broad class of functions.** The GP is capable of modelling a broad class of linear and nonlinear functions, even with the most standard design choices.

ii) **Closed form expressions.** The computational benefits of the Gaussian distribution are manifested in the GP as well, and no approximative computations are required in the inference.

iii) **Flexible.** The model complexity increases with the size of the data set, without the inclusion of more free parameters. Hence, the model is in a sense defined by the training data.

iv) **Non-parametric.** All measured data is required at the inference step. This differ from parametric models, which rely upon an explicit function ansatz in which the free parameters are chosen before the inference is performed. In the GP, one can view these parameters as absorbed by the model. What remains are the so called *hyperparparameters* –

that are a part of the *prior* distribution over the function – which are typically few in number.

v) **No manual tuning required.** While some regression algorithms require a rather arbitrary and manual parameter tuning, there are systematic, data-driven approaches for selecting the free hyperparparameters of the GP.

vi) **Uncertainty estimation.** Since the inference provides an entire distribution over the function, it includes the uncertainty of the model. Whereas this is not a property of the data, it provides a guideline on what parts of the input space where the model is struggling to perform well.

This being said, all that glitters is not gold; the main drawback of the GP is the poor scaling properties, which are cubic in time and quadratic in memory. Many approximation methods have been developed to overcome this obstacle, of which several rely upon so called *inducing variables* [26]; in this thesis we make use of a particularly well-suited basis function expansion [30].

Furthermore, the performance of the GP is dependent on the model specification, or the design of the prior distribution. Embedded user assumptions that are incorrect in important aspects hence limit the capacity of the method. To that end, there are a wide range of more or less complex constructions that extends the capabilities of the standard design choices; these include *deep kernel learning* [32] or *manifold GPs* [5], which we consider in this thesis in the light of integral measurements.

The aim of this chapter is to introduce the GP basics and give an overview of the design procedure. More advanced topics are discussed in Chapter 3.

2.1 A function as a random variable

Suppose that we are investigating a quantity mathematically described by the function

$$\mathtt{f}(\mathbf{x}) : \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}, \tag{2.1}$$

where the input is of dimension $D_\mathbf{x}$ while the output is scalar. We refer to $\mathtt{f}(\mathbf{x})$ as the *target function.*

Our goal is to build an approximation of the target function. To that end, assume that we have access to a set of measurements

$$y_i = \mathtt{f}(\mathbf{x}_i) + \epsilon_i, \qquad i = 1, \ldots, N, \tag{2.2}$$

where ϵ_i denotes the measurement noise. We use the indexing for sorting; y_i denotes measurement number i, collected at the input location $\mathbf{x}_i$.

Of course, we can not work with the target function $\mathtt{f}(\mathbf{x})$ directly, since it is unknown. Instead, we model $\mathtt{f}(\mathbf{x})$ with the GP $\mathscr{f}(\mathbf{x})$ [1]; expressed in terms of this model, the measurements are generated as

$$y_i = \mathscr{f}(\mathbf{x}_i) + \varepsilon_i, \qquad i = 1, \ldots, N. \tag{2.3}$$

Here, the modelled measurement noise ε_i is a realisation from the Gaussian distribution $\mathcal{N}\left(0,\ \sigma^2\right)$, where σ denotes the standard deviation. In many situations, this is a reasonable approximation of the real settings.

The GP models *the function value* $\mathscr{f}(\bar{\mathbf{x}})$ *at any point* $\bar{\mathbf{x}}$ *as a Gaussian random variable.* Hence, $\mathscr{f}(\bar{\mathbf{x}})$ is assigned a Gaussian *prior* distribution with the associated probability density $p(f(\bar{\mathbf{x}}))$, where $f(\bar{\mathbf{x}})$ lies in the sample space of $\mathscr{f}(\bar{\mathbf{x}})$. This model may appear rather abstract at the first encounter. Why does it make sense to treat the target function values as random variables? The answer is *uncertainty.* We use the prior distribution to encode our belief about the target function *before* the measurements have been observed; with no uncertainty, the prior distribution collapses to a point mass located at the target function value, such that $p(f(\bar{\mathbf{x}})) = \delta(f(\bar{\mathbf{x}}) - \mathtt{f}(\bar{\mathbf{x}}))$.

An illustration of the idea is given in Figure 2.1. The target function is taken to be $\mathtt{f}(x) = \sin x$, which is plotted in the horizontal (blue) plane. Consider the function value $\mathtt{f}(3\pi/2) = -1$, which we model with the Gaussian random variable $\mathscr{f}(3\pi/2)$. The vertical slice shows the probability density $p(f(3\pi/2))$ of the prior distribution $\mathcal{N}\left(0,\ 1\right)$. Under this prior, the probability density of the true function value is

$$p(\mathscr{f}(3\pi/2) = -1) = \mathcal{N}\left(-1;\ 0,\ 1\right) = 0.24.$$

Note that the prior distribution is a design choice, and we discuss it in more detail in Section 2.2.

Since any finite set of function values in the GP is just a collection of Gaussian random variables, it follows that they have a joint Gaussian distribution. A particularly important collection of variables is that of all function values at the input locations, which we model with the joint vector

$$\boldsymbol{\mathscr{f}} = \begin{bmatrix} \mathscr{f}(\mathbf{x}_1) \\ \vdots \\ \mathscr{f}(\mathbf{x}_N) \end{bmatrix}. \tag{2.4}$$

[1] One may note that in the literature – including the papers attached in this thesis – the target function $\mathtt{f}(\mathbf{x})$ and the model $\mathscr{f}(\mathbf{x})$ are often not distinguished in notation.

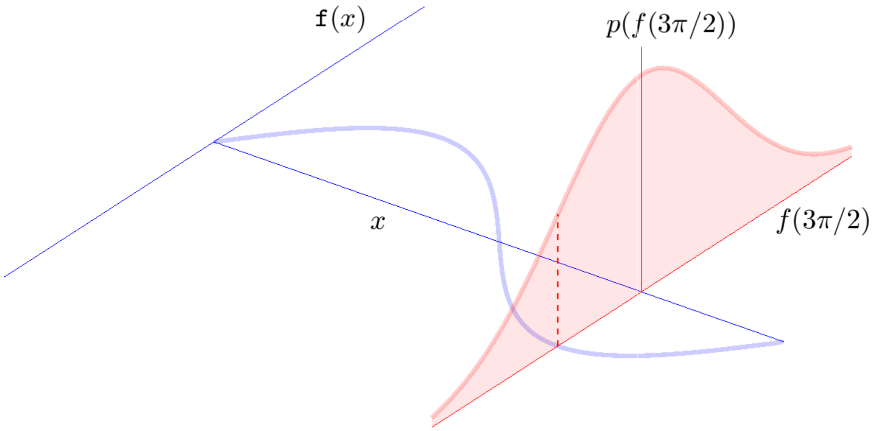


Figure 2.1: Illustration of the GP idea. Our uncertainty about the target function $\mathtt{f}(x)$ is encoded by treating its value in any point as a Gaussian random variable, which is defined by a prior distribution. Here, the target function is $\mathtt{f}(x) = \sin x$, and we have visualised the prior distribution of the random variable $f(3\pi/2)$.

The prior distribution of $\boldsymbol{f}$ is given by

$$\boldsymbol{f} \sim \mathcal{N}\left(\boldsymbol{\mu},\ K\right), \tag{2.5a}$$

$$\mu_i = \mathbb{E}[f(\mathbf{x}_i)], \tag{2.5b}$$

$$K_{ij} = \operatorname{Cov}\left[f(\mathbf{x}_i),\ f(\mathbf{x}_j)\right]. \tag{2.5c}$$

This particular covariance matrix K is called the *Gram matrix*. The entry K_{ij} specifies the covariance between $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$, and is determined by the *covariance function*

$$k(\mathbf{x}_i, \mathbf{x}_j) \triangleq \operatorname{Cov}\left[f(\mathbf{x}_i),\ f(\mathbf{x}_j)\right]. \tag{2.6}$$

Similarly, the mean value $\boldsymbol{\mu}$ is determined by the *mean function*

$$m(\mathbf{x}_i) \triangleq \mathbb{E}[f(\mathbf{x}_i)]. \tag{2.7}$$

Together, the mean function and the covariance function *uniquely* specify the GP; we denote this with the notation

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}),\ k(\mathbf{x}, \mathbf{x}')\right). \tag{2.8}$$

The covariance function is the most important design choice, as it specifies the amount of similarity the model assigns to different function values; specific choices for $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ are discussed in Section 2.2. Observe that the definitions easily allow us to extract the *marginal* distribution

$$f(\bar{\mathbf{x}}) \sim \mathcal{N}\left(m(\bar{\mathbf{x}}),\ k(\bar{\mathbf{x}}, \bar{\mathbf{x}})\right). \tag{2.9}$$

Relate this to the example illustrated by Figure 2.1, where we consider the function value at $\bar{x} = 3\pi/2$, using $m(\bar{x}) = 0$ and $k(\bar{\mathbf{x}}, \bar{\mathbf{x}}) = 1$.

Let us now define the measurement vector

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}, \tag{2.10}$$

which in our model is a realisation of the random variable

$$\boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu},\ K + \sigma^2 I\right). \tag{2.11}$$

Our goal is to predict the target function values at a set of test points $\{\mathbf{x}_{*i}\}_{i=1}^{N_*}$. These function values are modelled by

$$\boldsymbol{f}_* = \begin{bmatrix} f(\mathbf{x}_{*1}) \\ \vdots \\ f(\mathbf{x}_{*N_*}) \end{bmatrix}, \tag{2.12}$$

and we wish to determine the distribution of the variable $\boldsymbol{f}_* | \boldsymbol{y} = \mathbf{y}$, where we have conditioned $\boldsymbol{f}_*$ on the observed realisation $\boldsymbol{y} = \mathbf{y}$. To that end, we first note that the joint distribution of $\boldsymbol{y}$ and $\boldsymbol{f}_*$ is given by

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{bmatrix}, \begin{bmatrix} K + \sigma^2 I & K_* \\ K_*^\mathsf{T} & K_{**} \end{bmatrix} \right), \tag{2.13}$$

where

$$\boldsymbol{\mu}_* = \begin{bmatrix} m(x_{*1}) \\ \vdots \\ m(x_{*N_*}) \end{bmatrix}, \tag{2.14a}$$

$$K_* = \begin{bmatrix} k(x_1, x_{*1}) & \cdots & k(x_1, x_{*N_*}) \\ \vdots & \ddots & \vdots \\ k(x_N, x_{*1}) & \cdots & k(x_N, x_{*N_*}) \end{bmatrix}, \tag{2.14b}$$

and

$$K_{**} = \begin{bmatrix} k(x_{*1}, x_{*1}) & \cdots & k(x_{*1}, x_{*N_*}) \\ \vdots & \ddots & \vdots \\ k(x_{*N_*}, x_{*1}) & \cdots & k(x_{*N_*}, x_{*N_*}) \end{bmatrix}. \tag{2.14c}$$

Hence, K_* is the cross-covariance between $\boldsymbol{y}$ and $\boldsymbol{f}_*$ while K_{**} is the covariance of $\boldsymbol{f}_*$. By employing the conditional properties in a multivariate Gaussian distribution (see e.g [21]), we obtain

$$\begin{aligned} \boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y} &\sim \mathcal{N}\left(\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}],\ \mathrm{Cov}\left[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}\right]\right), \qquad (2.15a)\\ \mathbb{E}[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}] &= \boldsymbol{\mu}_* + K_*^{\mathsf{T}}(K + \sigma^2 I)^{-1}(\mathbf{y} - \boldsymbol{\mu}), \qquad (2.15b)\\ \mathrm{Cov}\left[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}\right] &= K_{**} - K_*^{\mathsf{T}}(K + \sigma^2 I)^{-1}K_*. \qquad (2.15c) \end{aligned}$$

This defines the *predictive distribution*, which in contrast to the prior encodes our belief about the target function *after* the measurements have been observed; the mean value $\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}]$ serves as our prediction of the target function at the test points. The analytically tractable closed form expressions explain, to a large extent, the popularity of the GP.

Due to the construction and inversion of an $N \times N$ matrix, GP regression scales as $\mathcal{O}\left(N^2\right)$ in memory and $\mathcal{O}\left(N^3\right)$ in time, making it naturally unfeasible for large data sets. As always, the computations shall not be done with explicit matrix inversion. A numerically stable technique is given by the Cholesky decomposition $LL^{\mathsf{T}} = K + \sigma^2 I$, where L is triangular; expressions of the form $(K + \sigma^2 I)^{-1}\mathbf{B}$ are then computed as the solution to $LL^{\mathsf{T}}\mathbf{A} = \mathbf{B}$, which is achieved through forward and backward substitutions.

Note that the entire covariance matrix $\mathrm{Cov}\left[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}\right]$ is useful mainly when drawing samples from this distribution; usually one is just interested in the variance $\mathrm{Var}[f(\mathbf{x}_{*i})]$ of each component, and hence it is enough to compute the diagonal part. It is important to understand that the computed variance is a property of the model and *not* a property of the data. In visualisations, it is not uncommon that the prediction is accompanied by what is claimed to be a *confidence region*, whereas the correct term is *credible region*. The distinction is important, since the credible region leaves no statistical guarantees of including the true function with any probability. This uncertainty estimate should therefore be interpreted with care.

A simple example of GP regression is seen in Figure 2.2, where 10 noisy measurements are generated from the sine function $\mathtt{f}(x) = \sin x$. We use a zero mean function $m(x) = 0$ and the squared exponential covariance function (see Section 2.2.1), both of which are common design choices. Note that the prediction reverts to 0 further away from the data, where no information is provided.

2.2 Mean and covariance functions

We mentioned earlier that the mean function $m(\mathbf{x})$ and the covariance function $k(\mathbf{x}, \mathbf{x}')$ *uniquely* specify the GP, and they are therefore our only design choices.

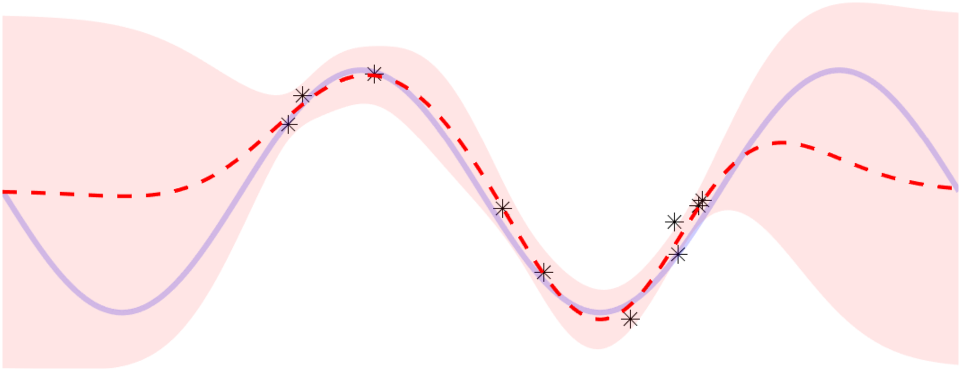


Figure 2.2: Illustration of GP regression, using 10 noisy measurements (stars) from a sine function (solid line). We show the prediction (dashed line) and the 95 % credible region. Further away from the data, the prediction is poor and reverts to the prior mean.

Their different roles are, if not otherwise, indicated by their names: the mean function specifies the prior belief of the function values, while the covariance function specifies the prior belief of the correlation between the function values. Put differently: the mean function is a guess of what the function is, while the covariance function is a guess of how the function behaves.

Designing the mean and covariance function is equivalent to designing the prior distribution of $f(\mathbf{x})$, and this is what GP modelling is all about; the closed forms in (2.15) make it easy to compute the predictive distribution – *the challenge lies in the design of the prior distribution.*

The mean function is the least important component of the two. In fact, a common design choice is to use a zero mean function $m(\mathbf{x}) \equiv 0$. One might incorrectly interpret this choice as if the target function is assumed to be 0 everywhere – it is rather a natural guess in the absence of any specific prior knowledge. However, the mean function could in theory by of any parameterised form.

The covariance function, on the other hand, is crucial. It stipulates the properties that we assign to the model $f(\mathbf{x})$, and should hence as well as possible reflect the properties of the target function. A mismatch between the behaviour we assign to the model and the true behaviour of the target function is likely to result in a poor predictive performance.

More loosely speaking, the covariance function determines the amount of similarity between different function values. The standard class of covariance functions are *stationary*, which means that they depend only on the difference $\mathbf{x} - \mathbf{x}'$. A covariance function dependent on the distance $\|\mathbf{x} - \mathbf{x}'\|_2$ is called *isotropic*. Stationarity is intuitive, since it implies that function values corresponding to nearby input locations are modelled as similar. However, and for the same reason, discontinuities in the target function are poorly modelled by a GP with a stationary covariance function; the classic example is a step function, where two points just on different sides of the step differ significantly in their function values. In such case, a non-stationary covariance functions is preferable.

In contrast to the mean function, the covariance function can not be constructed arbitrary but is subject to one crucial requirement: the resulting covariance matrix K must be positive semi-definite. If not, it is not a valid covariance matrix. It follows that K is symmetric, and hence that $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$.

Below we describe the two most common stationary covariance functions.

2.2.1 Squared exponential covariance function

The most well-known covariance function, not just of the stationary class, is the *squared exponential*

$$k_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left[-\frac{1}{2} l^{-2} \|\mathbf{x} - \mathbf{x}'\|_2^2\right], \tag{2.16}$$

where the magnitude parameter σ_f controls the variance of the process, and the parameter l is called *lengthscale*. The lengthscale impact how quickly the process tend to change, or how similar the function values are modelled to be – the extremes $l = 0$ and $l = \infty$ corresponds to a white noise process and a constant process, respectively.

The popularity of this covariance function is to a large extent explained by its simplicity – it is easy to implement and transform, while at the same time is often reasonably realistic. The squared exponential heavily favours smooth functions; in fact, a function governed by this process is infinitely differentiable [27].

Considering the one-dimensional case, it is particularly easy to relate this covariance function to a parametric model. Using the Maclaurin expansion of the exponential

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \ldots, \tag{2.17}$$

we note that

$$\begin{aligned} k_{\mathrm{SE}}(x, x') &= \sigma_f^2 e^{-\frac{1}{2}l^{-2}(x-x')^2} = \sigma_f^2 e^{-\frac{1}{2}l^{-2}x^2} e^{l^{-2}xx'} e^{-\frac{1}{2}l^{-2}x'^2} \\ &= \sigma_f^2 e^{-\frac{1}{2}l^{-2}x^2} \left[1 + l^{-2}xx' + \frac{1}{2}l^{-4}x^2x'^2 + \frac{1}{6}l^{-6}x^3x'^3 + \ldots\right] e^{-\frac{1}{2}l^{-2}x'^2} \\ &= \sigma_f^2 \mathscr{g}(x)^{\mathsf{T}} \mathscr{g}(x'), \end{aligned} \tag{2.18}$$

where

$$\boldsymbol{g}(x) = e^{-\frac{1}{2}l^{-2}x^2}\begin{bmatrix}1 & l^{-1}x & \frac{1}{\sqrt{2}}l^{-2}x^2 & \frac{1}{\sqrt{6}}l^{-3}x^3 & \dots\end{bmatrix}^{\mathsf{T}}. \tag{2.19}$$

Let us define the parameter vector

$$\boldsymbol{w} = \begin{bmatrix}w_1 & w_2 & w_3 & \dots\end{bmatrix}^{\mathsf{T}}, \tag{2.20}$$

and make the parametric ansatz

$$\boldsymbol{f}(x) = \boldsymbol{g}(x)^{\mathsf{T}}\boldsymbol{w}, \tag{2.21}$$

which we recognise as an infinite basis function expansion. By placing a Gaussian prior on the parameters

$$\boldsymbol{w} \sim \mathcal{N}\left(\mathbf{0},\ \sigma_f^2 I\right), \tag{2.22}$$

we find that

$$\begin{aligned}\operatorname{Cov}\left[\boldsymbol{f}(x),\ \boldsymbol{f}(x')\right] &= \mathbb{E}\left[\boldsymbol{f}(x)\boldsymbol{f}(x')^{\mathsf{T}}\right] = \mathbb{E}\left[\boldsymbol{g}(x)^{\mathsf{T}}\boldsymbol{w}\boldsymbol{w}^{\mathsf{T}}\boldsymbol{g}(x')\right] \\ &= \boldsymbol{g}(x)^{\mathsf{T}}\mathbb{E}\left[\boldsymbol{w}\boldsymbol{w}^{\mathsf{T}}\right]\boldsymbol{g}(x') = \boldsymbol{g}(x)^{\mathsf{T}}\sigma_f^2 I\boldsymbol{g}(x') \\ &= \sigma_f^2\boldsymbol{g}(x)^{\mathsf{T}}\boldsymbol{g}(x'),\end{aligned} \tag{2.23}$$

which equals $k_{\mathrm{SE}}(x, x')$ according to (2.18). Hence, we deduce that a one-dimensional GP governed by the squared exponential covariance function corresponds to a parametric model with infinitely many parameters. The fact that the covariance function can encode the same information with just two parameters is quite impressive.

A more general form of (2.16) is given by

$$k_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left[-\frac{1}{2}\sum_{k=1}^{D_{\mathbf{x}}} l_k^{-2}(x_k - x_k')^2\right], \tag{2.24}$$

where individual lengthscales are used in each input dimension. This facilitates modelling of functions that change with different rates along the different coordinate axis.

2.2.2 The Matérn family

Another important member of the stationary class is the Matérn family of covariance functions

$$k_{\mathrm{Matérn}}(r) = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{l}\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}r}{l}\right), \tag{2.25}$$

where $r = \|\mathbf{x} - \mathbf{x}'\|_2$, $l > 0$ and K_ν is a modified Bessel function. The fixed parameter $\nu > 0$ controls the smoothness of the process; as $\nu \to \infty$, the squared exponential (2.16) is regained. Hence, the Matérn family is a generalisation that relaxes the extreme smoothness assumptions encoded by the squared exponential, and is thereby often more realistic when modelling real-world functions. The particular choice $\nu = 5/2$ is commonly used, for which (2.25) takes the form

$$k_{\text{Matérn}_{5/2}}(r) = \sigma_f^2 \left(1 + \frac{\sqrt{5}r}{l} + \frac{5r^2}{3l^2}\right) \exp\left(-\frac{\sqrt{5}r}{l}\right). \tag{2.26}$$

The Matérn$_{5/2}$ covariance function is employed for strain field modelling in Paper II, while a rougher process with $\nu = 1$ is used for X-ray computed tomography reconstruction in Paper III.

A qualitative comparison between the squared exponential and the Matérn$_{5/2}$ covariance functions is seen in Figure 2.3, where five sample functions are drawn from the prior distribution of each process. The difference in smoothness is clear.

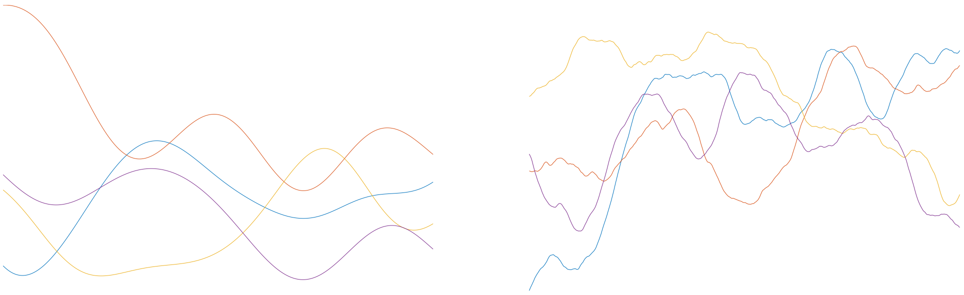


Figure 2.3: Sample functions drawn from processes governed by the squared exponential covariance function (left) and the Matérn$_{5/2}$ covariance function (right). While the former process is extremely smooth, the latter allows for more rough behaviours.

2.3 Hyperparameter selection

Having decided upon which mean and covariance function to use in our GP, the next thing we have to do is choose their free parameters. These are referred to as *hyperparameters*, to indicate that they are part of the prior distribution rather than of the target function itself. We use $\boldsymbol{\theta}$ to denote the set of hyperparameters $\{\theta_i\}$; in this set we include the noise standard deviation σ, since it is selected in the same procedure.

There are mainly two options available to this end: the first relies upon the *marginal likelihood* while the other employs *cross validation*. Both of these methods result in optimisation problems, but differ in their respective cost functions and the reasoning that lies behind them. Moreover, they both make use of the measured data $\mathbf{y}$, which might appear slightly strange: we have previously described the prior distribution as encoding our belief about the target function before the measurements are observed, yet we are still using them to design this distribution! A fully probabilistic approach models the hyperparameters as random variables and assign prior distributions to them as well; however, no approach of this kind has managed to compete with the mentioned methods in popularity.

The selection of the hyperparameters is by far the most time consuming part of GP regression; the cost function is typically more expensive to compute than the prediction itself, and several iterations are usually required to optimise it.

2.3.1 Marginal likelihood

Recall the Gaussian distribution of the measurements

$$\boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu},\ K + \sigma^2 I\right), \tag{2.27}$$

which was given in (2.11). The *marginal likelihood* is the value of the associated probability density function under the observation $\boldsymbol{y} = \mathbf{y}$, namely

$$p_{\boldsymbol{\theta}}(\mathbf{y}) = (2\pi)^{-\frac{N}{2}} \det(K + \sigma^2 I)^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^{\mathsf{T}}(K+\sigma^2 I)^{-1}(\mathbf{y}-\boldsymbol{\mu})}, \tag{2.28}$$

where we explicitly indicate the dependence on the hyperparameters. The idea is to select $\boldsymbol{\theta}$ so as to maximise $p_{\boldsymbol{\theta}}(\mathbf{y})$, based on the simple intuition of designing the prior such that our observed data becomes as likely as possible (given the choice of mean and covariance function). In practice, it is common to work with the logarithm of (2.28)

$$\begin{aligned} \log p_{\boldsymbol{\theta}}(\mathbf{y}) = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log\left[\det(K + \sigma^2 I)\right] \\ - \frac{1}{2}(\mathbf{y} - \boldsymbol{\mu})^{\mathsf{T}}(K + \sigma^2 I)^{-1}(\mathbf{y} - \boldsymbol{\mu}), \end{aligned} \tag{2.29}$$

where the goal is to find

$$\boxed{\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p_{\boldsymbol{\theta}}(\mathbf{y}).} \tag{2.30}$$

Some further insights are provided if we consider an alternative way of arriving at the expression (2.28), starting from *Bayes' theorem*

$$p_{\boldsymbol{\theta}}(\mathbf{f}|\boldsymbol{y}=\mathbf{y}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{y}|\boldsymbol{f}=\mathbf{f})p_{\boldsymbol{\theta}}(\mathbf{f})}{p_{\boldsymbol{\theta}}(\mathbf{y})}. \tag{2.31}$$

We refer to $p_{\boldsymbol{\theta}}(\mathbf{f}|\boldsymbol{y}=\mathbf{y})$ as the *posterior*. The right hand side includes the *likelihood*

$$p_{\boldsymbol{\theta}}(\mathbf{y}|\boldsymbol{f}=\mathbf{f}) = \mathcal{N}\left(\mathbf{y};\ \mathbf{f},\ \sigma^2 I\right), \tag{2.32}$$

while we recognise $p_{\boldsymbol{\theta}}(\mathbf{f}) = \mathcal{N}\left(\mathbf{f};\ \boldsymbol{\mu},\ K\right)$ as the prior. Integrating over $\mathbf{f}$ we note that

$$\int_{\mathbb{R}^N} p_{\boldsymbol{\theta}}(\mathbf{f}|\boldsymbol{y}=\mathbf{y})\,d\mathbf{f} = 1, \tag{2.33}$$

by the definition of a probability density. Since $p_{\boldsymbol{\theta}}(\mathbf{y})$ is independent of $\mathbf{f}$, (2.31) can be re-arranged to obtain

$$p_{\boldsymbol{\theta}}(\mathbf{y}) = \int_{\mathbb{R}^N} p_{\boldsymbol{\theta}}(\mathbf{y}|\boldsymbol{f}=\mathbf{f})p_{\boldsymbol{\theta}}(\mathbf{f})\,d\mathbf{f} = \int_{\mathbb{R}^N} \mathcal{N}\left(\mathbf{y};\ \mathbf{f},\ \sigma^2 I\right)\mathcal{N}\left(\mathbf{f};\ \boldsymbol{\mu},\ K\right)\,d\mathbf{f}. \tag{2.34}$$

Carrying out this integration, we get

$$\begin{aligned} p_{\boldsymbol{\theta}}(\mathbf{y}) &= c\int_{\mathbb{R}^N} e^{-\frac{1}{2}\sigma^{-2}(\mathbf{y}-\mathbf{f})^{\mathsf{T}}(\mathbf{y}-\mathbf{f})}e^{-\frac{1}{2}(\mathbf{f}-\boldsymbol{\mu})^{\mathsf{T}}K^{-1}(\mathbf{f}-\boldsymbol{\mu})}\,d\mathbf{f} \\ &= c\int_{\mathbb{R}^N} e^{-\frac{1}{2}\sigma^{-2}[\mathbf{y}^{\mathsf{T}}\mathbf{y}-2\mathbf{y}^{\mathsf{T}}\mathbf{f}+\mathbf{f}^{\mathsf{T}}\mathbf{f}]-\frac{1}{2}[\mathbf{f}^{\mathsf{T}}K^{-1}\mathbf{f}-2\boldsymbol{\mu}^{\mathsf{T}}K^{-1}\mathbf{f}+\boldsymbol{\mu}^{\mathsf{T}}K^{-1}\boldsymbol{\mu}]}\,d\mathbf{f} \\ &= c\int_{\mathbb{R}^N} e^{-\frac{1}{2}(\mathbf{f}-P^{-1}\mathbf{z})^{\mathsf{T}}P(\mathbf{f}-P^{-1}\mathbf{z})}\,d\mathbf{f}\ e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^{\mathsf{T}}(K+\sigma^2 I)^{-1}(\mathbf{y}-\boldsymbol{\mu})}, \end{aligned} \tag{2.35}$$

where

$$c = (2\pi)^{-N}\det(\sigma^{-2}K)^{-1/2}, \tag{2.36}$$

$$P = \sigma^{-2}I + K^{-1}, \tag{2.37}$$

$$\mathbf{z} = \sigma^{-2}\mathbf{y} + K^{-1}\boldsymbol{\mu}, \tag{2.38}$$

and we have used the identity $(\sigma^{-2}I + K^{-1})^{-1}\sigma^{-2} + \sigma^{-2}I = (K+\sigma^2 I)^{-1}$. Note that the last integrand in (2.35) is the un-normalised Gaussian density

$\mathcal{N}\left(\mathbf{f};\ P^{-1}\mathbf{z},\ P^{-1}\right)$, and the integral hence equals the inverse of the corresponding normalisation constant $(2\pi)^{-N/2}\det(P^{-1})^{-1/2}$. Since

$$\begin{aligned} c\cdot[(2\pi)^{-N/2}\det(P^{-1})^{-1/2}]^{-1} &= (2\pi)^{-N}\det(\sigma^{-2}K)^{-1/2}(2\pi)^{N/2}\det(P)^{-1/2} \\ &= (2\pi)^{-N/2}\det(\sigma^{2}KP)^{-1/2} \\ &= (2\pi)^{-N/2}\det(\sigma^{2}K(\sigma^{-2}I+K^{-1}))^{-1/2} \\ &= (2\pi)^{-N/2}\det(K+\sigma^{2}I)^{-1/2}, \end{aligned} \tag{2.39}$$

we get

$$p_{\boldsymbol{\theta}}(\mathbf{y}) = (2\pi)^{-\frac{N}{2}}\det(K+\sigma^2 I)^{-\frac{1}{2}}e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^{\mathsf{T}}(K+\sigma^2 I)^{-1}(\mathbf{y}-\boldsymbol{\mu})}, \tag{2.40}$$

which confirms the expression (2.28). This derivation explains the name *marginal* likelihood, since the integration is also referred to as *marginalisation* over $\mathbf{f}$.

2.3.2 Cross validation

Another common method in selecting the hyperparameters is to employ cross validation. The idea is to divide the data into two subsets, one *training* set and one *test* set. The training data is used to form a predictive distribution, under which the likeliness of the test data is computed. We obtain the cost function by repeating this procedure and collecting all contributions. The division into training and test sets reduces the risk of overfitting, since a perfect fit on the training data is likely to yield a bad fit on the test data. A special case of this approach is when the test data consist of one single measurement and the procedure is repeated for all N data points; this is referred to as *leave-one-out* cross validation.

Let $\tilde{\mathbf{y}}_i$ denote the measurement vector $\mathbf{y}$ with entry y_i removed. By setting $\boldsymbol{f}_* = f(\mathbf{x}_i)$ we can use (2.15) to compute

$$p_{\boldsymbol{\theta}}(y_i|\boldsymbol{y} = \tilde{\mathbf{y}}_i) = \mathcal{N}\left(y_i;\ \underbrace{m(\mathbf{x}_i) + \mathbf{k}_i^{\mathsf{T}}\hat{K}_i^{-1}(\tilde{\mathbf{y}}_k - \hat{\boldsymbol{\mu}}_k)}_{m_i},\ \underbrace{k(\mathbf{x}_i,\mathbf{x}_i) - \mathbf{k}_i^{\mathsf{T}}\hat{K}_i^{-1}\mathbf{k}_i}_{v_i^2}\right), \tag{2.41}$$

where $\hat{K}_i$ denotes K excluding row i and column i, $\hat{\boldsymbol{\mu}}_k$ denotes $\boldsymbol{\mu}$ except μ_i and

$$\mathbf{k}_i = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_i) & \cdots & k(\mathbf{x}_{i-1},\mathbf{x}_i) & k(\mathbf{x}_{i+1},\mathbf{x}_i) & \cdots k(\mathbf{x}_N,\mathbf{x}_N)\end{bmatrix}^{\mathsf{T}}. \tag{2.42}$$

The computational overhead is significantly reduced by noting that $\mathpzc{m}_i$ and $\mathpzc{v}_i^2$ can be efficiently obtained as

$$\mathpzc{v}_i^2 = 1/Z_{ii}, \tag{2.43a}$$

$$\mathpzc{m}_i = m(\mathbf{x}_i) + y_i - [Z(\mathbf{y} - \boldsymbol{\mu})]_i \mathpzc{v}_i^2, \tag{2.43b}$$

where $Z = (K + \sigma^2)^{-1}$. We construct the total contribution $\mathpzc{p}_{\boldsymbol{\theta}}(\mathbf{y})$ by multiplying the individual components

$$\mathpzc{p}_{\boldsymbol{\theta}}(\mathbf{y}) = \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(y_i | \boldsymbol{\mathpzc{y}} = \tilde{\mathbf{y}}_i). \tag{2.44}$$

As with the marginal likelihood, we prefer to work with the logarithmic form

$$\log \mathpzc{p}_{\boldsymbol{\theta}}(\mathbf{y}) = \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(y_i | \boldsymbol{\mathpzc{y}} = \tilde{\mathbf{y}}_i). \tag{2.45}$$

Note that since $p_{\boldsymbol{\theta}}(y_i | \boldsymbol{\mathpzc{y}} = \tilde{\mathbf{y}}_i)$ is a one-dimensional Gaussian density, the explicit logarithmic form is given as

$$\log p_{\boldsymbol{\theta}}(y_i | \boldsymbol{\mathpzc{y}} = \tilde{\mathbf{y}}_i) = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log \mathpzc{v}^2 - \frac{(y_i - \mathpzc{m}_i)^2}{2\mathpzc{v}_i^2}. \tag{2.46}$$

Conclusively, the desired hyperparameters are given by

$$\boxed{\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log \mathpzc{p}_{\boldsymbol{\theta}}(\mathbf{y}).} \tag{2.47}$$

2.3.3 Comparison and comments

As we have now seen, both ways of selecting the hyperparameters presented above result in optimisation problems. From a practical perspective, we should note that the partial derivatives of the cost functions (2.29) and (2.45) with respect to the hyperparameters are available in closed form, and hence a gradient based optimisation method is recommended. However, the cost functions are in general not concave and local maxima are likely to be present; this challenge is accentuated when the number of hyperparameters is large. Regarding computational complexity, the cross validation procedure is slightly more time consuming.

The two approaches stems from two quite different ways of thinking, with none of them necessarily being better than the other. It is fair to say that the marginal likelihood puts more trust in the model – we should stress that the intuition of designing the prior to maximise the likeliness of the data fails if the model assumptions are poor. In other words, if the model is unlikely to have generated the data in the first place, there is not much to win in making the data as likely as possible.

Cross validation does not evade the model assumptions, but relaxes them and weights the data higher; this reduces the sensitivity to model errors, but instead increases the risk of overfitting. An informal insight into the difference is obtained when considering the factorised expressions

$$p_{\boldsymbol{\theta}}(\mathbf{y}) = p_{\boldsymbol{\theta}}(y_1)p_{\boldsymbol{\theta}}(y_2|\boldsymbol{y}_1 = y_1)p_{\boldsymbol{\theta}}(y_3|\boldsymbol{y}_1 = y_1, \boldsymbol{y}_2 = y_2)\cdots , \tag{2.48}$$

for the marginal likelihod, and

$$\boldsymbol{p}_{\boldsymbol{\theta}}(\mathbf{y}) = p_{\boldsymbol{\theta}}(y_1|\boldsymbol{y} = \tilde{\mathbf{y}}_1)p_{\boldsymbol{\theta}}(y_2|\boldsymbol{y} = \tilde{\mathbf{y}}_2)p_{\boldsymbol{\theta}}(y_3|\boldsymbol{y} = \tilde{\mathbf{y}}_3)\cdots , \tag{2.49}$$

for cross validation. These are both products with N factors, but while every factor in the latter case conditions on *all* other data points, the former contains conditioning only on the *previous* data points. From this viewpoint, cross validation clearly values the data more than the marginal likelihood does.

Chapter 3

Extending the Gaussian process

In the previous chapter, we discussed the basic parts of GP modelling. This chapter considers some deepening topics.

Firstly, we extend the framework to multivariate target functions. Although this is straightforward notation-wise, it introduces a new design challenge; the covariance function becomes matrix-valued, which forces us to model the correlation between the output components.

Secondly, we study the behaviour of the GP as it undergoes linear functional transformations. It turns out that the resulting object is also a GP, in analogy with how a linear transformation of a Gaussian random variable preserves Gaussianity.

Thirdly, we consider a spectral based approximation method that makes the GP applicable for large data sets and allows for efficient computations of the covariance components.

Lastly, we review the concept of *deep kernel learning*, which is based on the idea of transforming the inputs to the covariance function through a neural network. This enables a more powerful approach when it comes to modelling functions with discontinuous features.

3.1 Multivariate Gaussian processes

So far in this thesis, we have limited our interest to a scalar target function. However, the GP framework is straightforward to generalise to target functions of arbitrary dimension $D_{\mathtt{F}}$

$$\mathtt{F}(\mathbf{x}) = \begin{bmatrix} \mathtt{f}_1(\mathbf{x}) \\ \vdots \\ \mathtt{f}_{D_{\mathtt{F}}}(\mathbf{x}) \end{bmatrix}. \tag{3.1}$$

We model $\mathtt{F}(\mathbf{x})$ with the GP

$$\mathcal{F}(\mathbf{x}) = \begin{bmatrix} \mathscr{f}_1(\mathbf{x}) \\ \vdots \\ \mathscr{f}_{D_{\mathtt{F}}}(\mathbf{x}) \end{bmatrix} \sim \mathcal{GP}\left(\mathbf{m}(\mathbf{x}),\ \mathbf{K}(\mathbf{x}, \mathbf{x}')\right), \tag{3.2}$$

where

$$\mathbf{m}(\mathbf{x}) = \begin{bmatrix} m_1(\mathbf{x}) \\ \vdots \\ m_{D_{\mathtt{F}}}(\mathbf{x}) \end{bmatrix}, \tag{3.3}$$

and

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \begin{bmatrix} k_{11}(\mathbf{x}, \mathbf{x}') & \cdots & k_{1D_{\mathtt{F}}}(\mathbf{x}, \mathbf{x}') \\ \vdots & \ddots & \vdots \\ k_{D_{\mathtt{F}}1}(\mathbf{x}, \mathbf{x}') & \cdots & k_{D_{\mathtt{F}}D_{\mathtt{F}}}(\mathbf{x}, \mathbf{x}') \end{bmatrix}. \tag{3.4}$$

The vector-valued mean function is a direct extension and does not introduce any new challenges, especially not when considering the common choice $\mathbf{m}(\mathbf{x}) \equiv \mathbf{0}$. Instead, the important difference lies in the matrix-valued covariance function. Apart from modelling the covariance of each output component, we now also have to model the correlation between them. This is a nontrivial design challenge that has gained a fair amount of interest; a useful review is given in [2].

The simplest option is to set $\mathbf{K}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')I$, where $k(\mathbf{x}, \mathbf{x}')$ is a scalar covariance function. With this choice, the components are modelled as independent while sharing the same covariance properties. A slight extension is to allow different scalar covariance functions for each component, which gives a pure independence structure. From a practical viewpoint, it is here beneficial to employ the vectorisation format

$$\left[\mathscr{f}_1(\mathbf{x}_1) \cdots \mathscr{f}_1(\mathbf{x}_N) \cdots\ \cdots \mathscr{f}_{D_{\mathtt{F}}}(\mathbf{x}_1) \cdots \mathscr{f}_{D_{\mathtt{F}}}(\mathbf{x}_N)\right]^{\mathsf{T}}. \tag{3.5}$$

The Gram matrix then becomes block-diagonal, which can be exploited for more efficient computations.

Although the simplicity is appealing and provides a straightforward implementation, the approach is unsatisfying since it neglects potentially valuable correlation information. Consequently, there exist several more sophisticated constructions. One example is given by the so called *separable kernels*, with the idea of expressing $\mathbf{K}(\mathbf{x}, \mathbf{x}')$ as a linear combination of scalar covariance functions

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \sum_i k_i(\mathbf{x}, \mathbf{x}') B_i. \tag{3.6}$$

The symmetric, positive semi-definite matrices B_i are found via some kind of regularisation. For instance, the *mixed effect regulariser* gives the form

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')\big(\omega \mathbf{1} + (1-\omega) I\big), \tag{3.7}$$

where $\mathbf{1}_{ij} = 1$ and ω is a tuning parameter determining the level of correlation: with $\omega = 0$ we regain the diagonal form, while $\omega = 1$ models the components as identical.

The class of *invariant kernels* is of particular interest for this thesis. Here, $\mathbf{K}(\mathbf{x}, \mathbf{x}')$ encodes physical background knowledge of the target function. This class include the curl and divergence free covariance functions [3, 20], which has been used for GP modelling of magnetic fields [29, 31]. The construction relies upon the existence of a *potential* function – a curl-free vector field is the gradient of a scalar potential while a divergence-free vector field is the curl of a vector potential. Modelling the underlying potential with a GP, the desired covariance function is derived in a fairly straightforward manner. The salient point is that the functions described by and generated from the resulting process are enforced to obey the encoded property; hence a curl-free covariance function does only allow curl-free functions. In this way, the regression is restricted with respect to the underlying physics.

In Paper I, we describe a practical procedure for the construction of invariant kernels provided linear constraints on the target function. A brief background to this method is given in Section 3.2.3.

Finally, we underline that the predictive expressions (2.15) remain unchanged; the multivariate extension simply correspond to replacing all scalar entries in the previously built matrices with their multivariate equivalents.

3.2 Linear functionals in Gaussian processes

In the model introduced in the previous chapter, we assumed that the measurements are noisy point-evaluations of the target function. However, this is not always the case. Several interesting problems involves measurements of derivatives or integrals of the target function. These operators belong to the class of linear functionals, and are surprisingly easy to include in the GP framework. This section explains how.

3.2.1 A linearly transformed Gaussian random variable

To begin with, let us briefly recall the impact of linear transformations in the Gaussian distribution. Let X be a Gaussian random variable with mean μ_X and standard deviation σ_X

$$X \sim \mathcal{N}\left(\mu_X,\ \sigma_X^2\right). \tag{3.8}$$

In one dimension, a linear transformation is a pure scaling that constructs a new variable $Z = aX$. Using the definitions of mean value and variance, it is easy to verify that Z is a Gaussian random variable with distribution

$$Z \sim \mathcal{N}\left(a\mu_X,\ (a\sigma_X)^2\right). \tag{3.9}$$

In other words, the mean value and standard deviation are scaled with the same factor as the random variable itself.

In the multivariate case, we consider

$$\mathbf{X} \sim \mathcal{N}\left(\boldsymbol{\mu}_\mathbf{X},\ \Sigma_X\right), \qquad \mathbf{X}, \boldsymbol{\mu}_\mathbf{X} \in \mathbb{R}^{D_\mathbf{X}}, \qquad \Sigma_X \in \mathbb{R}^{D_\mathbf{X} \times D_\mathbf{X}}, \tag{3.10}$$

and the linear transformation $\mathbf{Z} = A\mathbf{X}$, where A is a matrix. The distribution of $\mathbf{Z}$ reads

$$\mathbf{Z} \sim \mathcal{N}\left(A\boldsymbol{\mu}_\mathbf{X},\ A\Sigma_X A^\mathsf{T}\right). \tag{3.11}$$

These basic but important properties turn out to have equivalent formulations when considering linear functionals in GPs.

3.2.2 Scalar linear functionals

We use $\mathcal{L}$ to denote a linear functional. Being linear means that it fulfils the superposition principle

$$\mathcal{L}[a\mathtt{f}(\mathbf{x}) + b\mathtt{g}(\mathbf{x})] = a\mathcal{L}\mathtt{f}(\mathbf{x}) + b\mathcal{L}\mathtt{g}(\mathbf{x}), \tag{3.12}$$

for two functions $\mathtt{f}(\mathbf{x})$ and $\mathtt{g}(\mathbf{x})$, and scalars a and b. A functional is informally described as *a function of a function.* The most basic example, which seldom is referred to as a functional, is point evaluation

$$\mathcal{L}_{\bar{\mathbf{x}}}\mathtt{f}(\mathbf{x}) = \mathtt{f}(\bar{\mathbf{x}}), \tag{3.13}$$

which simply returns the function value at the point $\mathbf{x} = \bar{\mathbf{x}}$. Another common example is differentiation

$$\mathcal{L}_{\bar{\mathbf{x}},j}\mathtt{f}(\mathbf{x}) = \frac{\partial \mathtt{f}(\mathbf{x})}{\partial x_j}\Big|_{\mathbf{x}=\bar{\mathbf{x}}}, \tag{3.14}$$

while the third and final functional we consider is integration

$$\mathcal{L}_\Omega \mathtt{f}(\mathbf{x}) = \int_\Omega \mathtt{f}(\mathbf{x})\, d\mathbf{x}. \tag{3.15}$$

A special case of big importance for this thesis is the line integral

$$\mathcal{L}_\eta \mathtt{f}(\mathbf{x}) = \int_0^L \mathtt{f}(\bar{\mathbf{x}} + s\hat{\mathbf{n}})\, ds, \tag{3.16}$$

where $\eta = \{\bar{\mathbf{x}}, L, \hat{\mathbf{n}}\}$. The integration is here made along the straight line segment of length L between the points $\bar{\mathbf{x}}$ and $\bar{\mathbf{x}} + L\hat{\mathbf{n}}$, where the direction is specified by the unit vector $\hat{\mathbf{n}}$.

Given the GP

$$\mathscr{f}(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}),\ k(\mathbf{x}, \mathbf{x}')\right), \tag{3.17}$$

what can we say about $\mathcal{L}\mathscr{f}(\mathbf{x})$? In the same way as a linear transformation preserves Gaussianity of a Gaussian random variable, so does a linear functional preserves Gaussianity in the GP; this implies that $\mathcal{L}\mathscr{f}(\mathbf{x})$ is a GP as well [6, 13]. The linearity of $\mathcal{L}$ allows for a straightforward derivation of this process

$$\mathbb{E}[\mathcal{L}\mathscr{f}(\mathbf{x})] = \mathcal{L}\mathbb{E}[\mathscr{f}(\mathbf{x})] = \mathcal{L}m(\mathbf{x}), \tag{3.18a}$$

$$\mathrm{Cov}\left[\mathcal{L}\mathscr{f}(\mathbf{x}),\ \mathcal{L}'\mathscr{f}(\mathbf{x}')\right] = \mathcal{L}\mathcal{L}'\mathrm{Cov}\left[\mathscr{f}(\mathbf{x}),\ \mathscr{f}(\mathbf{x}')\right] = \mathcal{L}\mathcal{L}'k(\mathbf{x}, \mathbf{x}'), \tag{3.18b}$$

where $\mathcal{L}'$ denotes the functional acting on the argument $\mathbf{x}'$. Hence, we conclude that

$$\boxed{\mathcal{L}\mathscr{f}(\mathbf{x}) \sim \mathcal{GP}\left(\mathcal{L}m(\mathbf{x}),\ \mathcal{L}\mathcal{L}'k(\mathbf{x}, \mathbf{x}')\right),} \tag{3.19}$$

which is the GP equivalent to (3.9).

Next, it is important to realise that the processes $\mathscr{f}(\mathbf{x})$ and $\mathcal{L}\mathscr{f}(\mathbf{x})$ are not separate from each other, an insight that follows from the fact that they are both collections of Gaussian random variables. Consider the generalised measurement model

$$y_i = \mathcal{L}_i\mathscr{f}(\mathbf{x}) + \varepsilon_i. \tag{3.20}$$

Here, we let the subscript i in $\mathcal{L}_i$ encode *both* the measurement index *as well as* the kind of functional and its input. Note that by setting $\mathcal{L}_i\mathscr{f}(\mathbf{x}) = \mathscr{f}(\mathbf{x}_i)$ for all i, we regain the point measurement model (2.3). The predictive expressions (2.15) remain of exactly the same form

$$\boxed{\begin{aligned} \boldsymbol{\ell}_* | \boldsymbol{y} = \mathbf{y} &\sim \mathcal{N}\left(\mathbb{E}[\boldsymbol{\ell}_* | \boldsymbol{y} = \mathbf{y}],\ \mathrm{Cov}\left[\boldsymbol{\ell}_* | \boldsymbol{y} = \mathbf{y}\right]\right), && (3.21a)\\ \mathbb{E}[\boldsymbol{\ell}_* | \boldsymbol{y} = \mathbf{y}] &= \tilde{\boldsymbol{\mu}}_* + Q_*^{\mathsf{T}}(Q + \sigma^2 I)^{-1}(\mathbf{y} - \tilde{\boldsymbol{\mu}}), && (3.21b)\\ \mathrm{Cov}\left[\boldsymbol{\ell}_* | \boldsymbol{y} = \mathbf{y}\right] &= Q_{**} - Q_*^{\mathsf{T}}(Q + \sigma^2 I)^{-1} Q_*, && (3.21c) \end{aligned}}$$

although we have replaced the previous quantities with the generalisations

$$\begin{aligned} \ell_{*i} &= \mathcal{L}_{*i} f(\mathbf{x}), && (3.22a)\\ \tilde{\mu}_i &= \mathcal{L}_i m(\mathbf{x}), && (3.22b)\\ \tilde{\mu}_{*i} &= \mathcal{L}_{*i} m(\mathbf{x}), && (3.22c)\\ Q_{ij} &= \mathcal{L}_i \mathcal{L}'_j k(\mathbf{x}, \mathbf{x}'), && (3.22d)\\ [Q_*]_{ij} &= \mathcal{L}_i \mathcal{L}'_{*j} k(\mathbf{x}, \mathbf{x}), && (3.22e)\\ [Q_{**}]_{ij} &= \mathcal{L}_{*i} \mathcal{L}'_{*j} k(\mathbf{x}, \mathbf{x}'). && (3.22f) \end{aligned}$$

What this means, more precisely, is that we can use a combination of different linear functional measurements and predict for any combination of linear functional evaluations.

Let us concretise this with an example, once again using the sine function $\mathtt{f}(x) = \sin x$. We collect measurements of a function value, a derivative, and an integral, all at different input locations; see the top left plot of Figure 3.1. The remaining plots are showing the mean values and 95 % credible regions when predicting with respect to the function value (top right), the derivative (bottom left) and the integral $\int_0^{x_*} \mathtt{f}(x)\, dx$ (bottom right). We are using the squared exponential covariance function (2.16). Note where the model uncertainty attains its minimum value in each case, and relate it to the corresponding measurement. Of course, all measurements are equally well obeyed in every prediction, although it is not directly revealed in the visualisations.

3.2.3 Multivariate functionals and exploitation of linear constraints

In Section 3.1, we briefly discussed *invariant kernels*, covariance functions for multivariate GPs that encode physical constraints. Here, we outline a method that generalises their construction. To that end, we first need to extend the concepts of the previous section to functionals with multiple outputs; one such example is gradient evaluations. In the general case, we consider a multivariate linear functional $\boldsymbol{\mathcal{G}}$ acting on a multivariate GP $\mathcal{T}(\mathbf{x})$, defined by the mean function $\mathbf{m}_{\mathcal{T}}(\mathbf{x})$ and covariance function $\mathbf{K}_{\mathcal{T}}(\mathbf{x}, \mathbf{x}')$ (we use $\mathcal{T}$

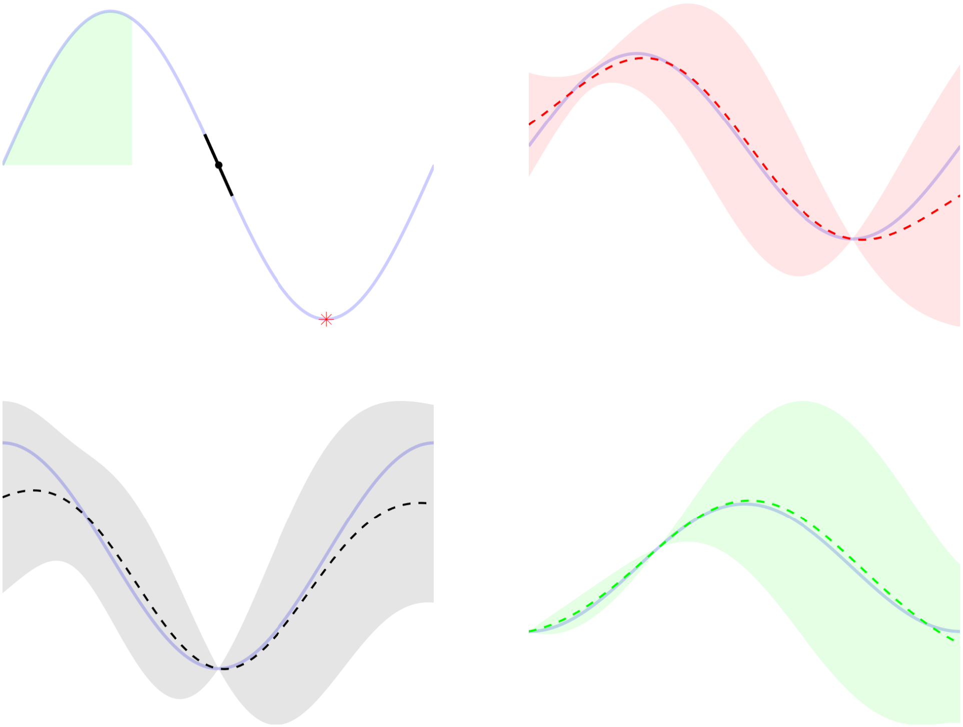


Figure 3.1: Illustration of the GPs ability of incorporating linear functionals. *Top left*: True function and measurements of a function value (red star), a derivative (black dot with tangent) and an integral (green shaded region). *Remaining plots*: predictive distributions with 95 % credible regions for the function (top right), the derivative (bottom left) and the integral $\int_0^{x_*} \mathtt{f}(x)\,dx$ (bottom right). True quantities are shown in solid blue, mean predictions are dashed.

instead of $\mathcal{F}$ to denote this GP, for reasons that will be clear below). We write this as $\boldsymbol{\mathfrak{G}}\mathcal{T}(\mathbf{x})$; due to the linearity, we can treat this as a matrix product where each entry in $\boldsymbol{\mathfrak{G}}$ encodes a scalar linear functional. In analogy to the scalar case, $\boldsymbol{\mathfrak{G}}\mathcal{T}(\mathbf{x})$ remains a GP and is defined by

$$\begin{aligned}
\mathbb{E}[\boldsymbol{\mathfrak{G}}\mathcal{T}(\mathbf{x})] &= \boldsymbol{\mathfrak{G}}\mathbb{E}[\mathcal{T}(\mathbf{x})] = \boldsymbol{\mathfrak{G}}\mathbf{m}_{\mathcal{T}}(\mathbf{x}), && (3.23a)\\
\operatorname{Cov}\left[\boldsymbol{\mathfrak{G}}\mathcal{T}(\mathbf{x}),\ \boldsymbol{\mathfrak{G}}'\mathcal{T}(\mathbf{x}')\right] &= \mathbb{E}[(\boldsymbol{\mathfrak{G}}\mathcal{T}(\mathbf{x}) - \boldsymbol{\mathfrak{G}}\mathbf{m}_{\mathcal{T}}(\mathbf{x}))(\boldsymbol{\mathfrak{G}}'\mathcal{T}(\mathbf{x}') - \boldsymbol{\mathfrak{G}}'\mathbf{m}_{\mathcal{T}}(\mathbf{x}'))^{\mathsf{T}}]\\
&= \mathbb{E}[\boldsymbol{\mathfrak{G}}(\mathcal{T}(\mathbf{x}) - \mathbf{m}_{\mathcal{T}}(\mathbf{x}))(\mathcal{T}(\mathbf{x}') - \mathbf{m}_{\mathcal{T}}(\mathbf{x}'))^{\mathsf{T}}\boldsymbol{\mathfrak{G}}'^{\mathsf{T}}]\\
&= \boldsymbol{\mathfrak{G}}\mathbb{E}[(\mathcal{T}(\mathbf{x}) - \mathbf{m}_{\mathcal{T}}(\mathbf{x}))(\mathcal{T}(\mathbf{x}') - \mathbf{m}_{\mathcal{T}}(\mathbf{x}'))^{\mathsf{T}}]\boldsymbol{\mathfrak{G}}'^{\mathsf{T}}\\
&= \boldsymbol{\mathfrak{G}}\operatorname{Cov}\left[\mathcal{T}(\mathbf{x})\right]\boldsymbol{\mathfrak{G}}'^{\mathsf{T}} = \boldsymbol{\mathfrak{G}}\mathbf{K}_{\mathcal{T}}(\mathbf{x},\mathbf{x}')\boldsymbol{\mathfrak{G}}'^{\mathsf{T}}. && (3.23b)
\end{aligned}$$

Note the similarity with (3.11); we summarise this extension as

$$\boxed{\boldsymbol{\mathscr{G}}\mathcal{T}(\mathbf{x}) \sim \mathcal{GP}\left(\boldsymbol{\mathscr{G}}\mathbf{m}_{\mathcal{T}}(\mathbf{x}),\ \boldsymbol{\mathscr{G}}\mathbf{K}_{\mathcal{T}}(\mathbf{x},\mathbf{x}')\boldsymbol{\mathscr{G}}'^{\mathsf{T}}\right).} \tag{3.24}$$

With this knowledge, we face the problem of designing the covariance function $\mathbf{K}(\mathbf{x},\mathbf{x}')$ of $\mathcal{F}(\mathbf{x})$, assuming that we have access to constraints on the target function of the form

$$\boldsymbol{\mathscr{F}}\mathtt{F}(\mathbf{x}) = \mathbf{0}. \tag{3.25}$$

For instance, a curl-free function fulfils the relation

$$\underbrace{\begin{bmatrix} 0 & -\frac{\partial}{\partial x_3} & \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} & 0 & -\frac{\partial}{\partial x_1} \\ -\frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_1} & 0 \end{bmatrix}}_{\boldsymbol{\mathscr{F}}} \underbrace{\begin{bmatrix} \mathtt{f}(\mathbf{x}_1) \\ \mathtt{f}(\mathbf{x}_2) \\ \mathtt{f}(\mathbf{x}_3) \end{bmatrix}}_{\mathtt{F}(\mathbf{x})} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{0}}. \tag{3.26}$$

We further assume that the target function is related to another function $\mathtt{T}(\mathbf{x})$ through the mapping $\boldsymbol{\mathscr{G}}$, such that $\mathtt{F}(\mathbf{x}) = \boldsymbol{\mathscr{G}}\mathtt{T}(\mathbf{x})$. This implies that our GP model $\mathcal{F}(\mathbf{x})$ of $\mathtt{F}(\mathbf{x})$ can be expressed as $\boldsymbol{\mathscr{G}}\mathcal{T}(\mathbf{x})$, where $\mathcal{T}(\mathbf{x})$ is a GP modelling $\mathtt{G}(\mathbf{x})$. Imposing the constraints (3.25) on our model yields

$$\boldsymbol{\mathscr{F}}\mathcal{F}(\mathbf{x}) = \mathbf{0} \quad \Leftrightarrow \quad \boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{G}}\mathcal{T}(\mathbf{x}) = \mathbf{0}. \tag{3.27}$$

We want this hold for an arbitrary $\mathcal{T}(\mathbf{x})$, which it does if

$$\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{G}} = \mathbf{0}. \tag{3.28}$$

Hence, we need to design $\boldsymbol{\mathscr{G}}$ such that $\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{G}} = \mathbf{0}$ is fulfilled; then $\mathcal{F}(\mathbf{x})$ is completely specified by (3.24). A constructive procedure for this design is given in Paper I. A subsequent, rigorous theoretical analysis of the existence and properties of this construction is given in [18].

3.3 Hilbert space approximation

The most obvious drawback of GP regression is the poor scaling, which as mentioned earlier is $\mathcal{O}\left(N^3\right)$ and $\mathcal{O}\left(N^2\right)$ in time and memory, respectively. For this reason, there is a big interest in approximation methods that make the GP tractable when N is large. In this section we consider one in particular that has been of great use in this thesis, originally proposed and thoroughly described in [30]. The method relies upon a Hilbert space approximation of the covariance function, and the result is a low rank approximation of the Gram matrix that enables reformulations of the predictive expressions for significantly more efficient computations.

Assume that we want to build a GP $f(\mathbf{x})$ governed by the stationary covariance function

$$k(\mathbf{x}, \mathbf{x}') = k(\mathbf{r}), \tag{3.29}$$

where $\mathbf{r} = \mathbf{x} - \mathbf{x}' = [r_1, \dots, r_{D_\mathbf{x}}]$. A key component in this method is the fact that stationary covariance functions have a dual relationship with their *spectral density* $S(\boldsymbol{\omega})$ as

$$k(\mathbf{r}) = \frac{1}{(2\pi)^{D_\mathbf{x}}} \int S(\boldsymbol{\omega}) e^{i\boldsymbol{\omega}^\mathsf{T}\mathbf{r}} \, d\boldsymbol{\omega}, \tag{3.30a}$$

$$S(\boldsymbol{\omega}) = \int k(\mathbf{r}) e^{-i\boldsymbol{\omega}^\mathsf{T}\mathbf{r}} \, d\mathbf{r}, \tag{3.30b}$$

where $\boldsymbol{\omega} = \mathbf{x} - \mathbf{x}' = [\omega_1, \dots, \omega_{D_\mathbf{x}}]$. For instance, the spectral density of the squared exponential covariance function (2.16) is given by

$$S_{\text{SE}}(\boldsymbol{\omega}) = \sigma_f^2 (2\pi l^2)^{D_\mathbf{x}/2} \exp\left[-\frac{1}{2} l^2 \|\boldsymbol{\omega}\|_2^2\right]. \tag{3.31}$$

We approximate $f(\mathbf{x})$ with another GP $f_\phi(\mathbf{x})$, formed through the finite basis function expansion

$$f_\phi(\mathbf{x}) = \sum_{i=1}^{m} \phi_i(\mathbf{x}) w_i = \boldsymbol{\phi}(\mathbf{x})^\mathsf{T} \boldsymbol{w}, \tag{3.32}$$

where

$$\boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} \phi_1(\mathbf{x}) \\ \vdots \\ \phi_m(\mathbf{x}) \end{bmatrix}, \qquad \text{and} \qquad \boldsymbol{w} = \begin{bmatrix} w_1(\mathbf{x}) \\ \vdots \\ w_m(\mathbf{x}) \end{bmatrix}. \tag{3.33}$$

Furthermore, we limit our interest in $f_\phi(\mathbf{x})$ to the finite domain $\Omega \in \mathbb{R}^{D_\mathbf{x}}$, which is taken to be rectangular such that

$$\Omega = [-L_1, L_1] \times \cdots \times [-L_{D_\mathbf{x}}, L_{D_\mathbf{x}}]. \tag{3.34}$$

The basis functions are obtained from the Laplace eigenvalue problem

$$\begin{cases} -\Delta\phi_j(\mathbf{x}) = \lambda_j \phi_j(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \phi_j(\mathbf{x}) = 0, & \mathbf{x} \in \partial\Omega, \end{cases} \tag{3.35}$$

where λ_j is the eigenvalue associated with basis function j. Note that the boundary condition in (3.35) is explicitly imposed on $f_\phi(\mathbf{x})$ as well, and hence the domain must be chosen large enough to prevent undesired impact on the prediction. The solution to (3.35) is found using separation of variables (see Appendix A), and is given by

$$\phi_j(\mathbf{x}) = \prod_{k=1}^{D_\mathbf{x}} L_k^{-1/2} \sin\left[c_{kj}(x_k + L_k)\right], \qquad c_{kj} = \frac{j_k \pi}{2L_k}, \qquad \lambda_j = \sum_{k=1}^{D_\mathbf{x}} c_{kj}^2. \tag{3.36}$$

We use the notation j_k to denote that basis function j has index k in direction k; see Figure 3.2 for an illustration. Moreover, the basis functions are orthonormal on Ω

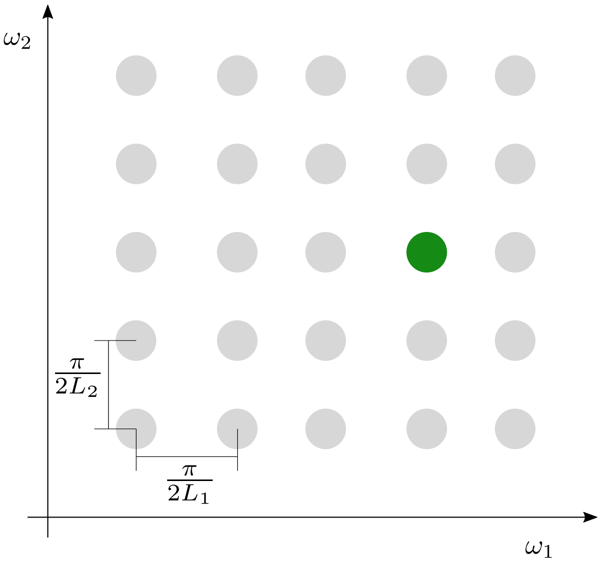


Figure 3.2: Illustration of the frequency components of the basis functions, considering two dimensions and a total number $m = 25$. Basis function j corresponding to the highlighted green circle encodes the 4th and 3rd frequency component in the first and second direction, respectively; therefore $j_1 = 4$ and $j_2 = 3$.

$$\int_\Omega \phi_i(\mathbf{x})\phi_j(\mathbf{x})\, d\mathbf{x} = \delta_{ij}. \tag{3.37}$$

The covariance function $k_\phi(\mathbf{x}, \mathbf{x}')$ of the approximate process $f_\phi(\mathbf{x})$ is

$$k_\phi(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\mathsf{T}\mathbb{E}[\boldsymbol{w}_m \boldsymbol{w}_m^\mathsf{T}]\boldsymbol{\phi}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\mathsf{T}\mathrm{Cov}\left[\boldsymbol{w}_m\right]\boldsymbol{\phi}(\mathbf{x}). \tag{3.38}$$

Letting $\mathbf{c}_j = [c_{1j}, \ldots, c_{D_\mathbf{x}}]$ and

$$\mathrm{Cov}\left[\boldsymbol{w}_m\right] = \Lambda = \begin{bmatrix} S(\mathbf{c}_1) & & \\ & \ddots & \\ & & S(\mathbf{c}_m) \end{bmatrix}, \tag{3.39}$$

it follows that

$$k_\phi(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{m} S(\mathbf{c}_j)\phi_j(\mathbf{x})\phi_j(\mathbf{x}'). \tag{3.40}$$

In the limit $m, L_k \to \infty$ it holds that $k(\mathbf{x}, \mathbf{x}') = k_\phi(\mathbf{x}, \mathbf{x}')$ [30]. This limit corresponds to a complete coverage of the entire spectral domain: imagine placing circles on *every* single point in Figure 3.2. In practise, we use finite values of m and L_k which yields the approximation $k(\mathbf{x}, \mathbf{x}') \approx k_\phi(\mathbf{x}, \mathbf{x}')$. By constructing the matrices

$$\Phi = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \cdots & \phi_m(\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ \phi_1(\mathbf{x}_N) & \cdots & \phi_m(\mathbf{x}_N) \end{bmatrix}, \tag{3.41}$$

$$\Phi_* = \begin{bmatrix} \phi_1(\mathbf{x}_{*1}) & \cdots & \phi_m(\mathbf{x}_{*1}) \\ \vdots & \vdots & \vdots \\ \phi_1(\mathbf{x}_{*N_*}) & \cdots & \phi_m(\mathbf{x}_{*N_*}) \end{bmatrix}, \tag{3.42}$$

we form the approximations

$$K \approx \Phi\Lambda\Phi^\mathsf{T}, \tag{3.43a}$$
$$K_* \approx \Phi\Lambda\Phi_*^\mathsf{T}, \tag{3.43b}$$
$$K_{**} \approx \Phi_*\Lambda\Phi_*^\mathsf{T}. \tag{3.43c}$$

Making these substitutions in the exact expressions (2.15), we obtain

$$\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}] \approx \boldsymbol{\mu}_* + \Phi_*\Lambda\Phi^\mathsf{T}(\Phi\Lambda\Phi^\mathsf{T} + \sigma^2 I)^{-1}(\mathbf{y} - \boldsymbol{\mu}), \tag{3.44a}$$
$$\mathrm{Cov}\left[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}\right] \approx \Phi_*\Lambda\Phi_*^\mathsf{T} - \Phi_*\Lambda\Phi^\mathsf{T}(\Phi\Lambda\Phi^\mathsf{T} + \sigma^2 I)^{-1}\Phi\Lambda\Phi_*^\mathsf{T}. \tag{3.44b}$$

Since the same matrix dimensions are maintained, the computational complexity is not reduced. However, by applying the matrix inversion lemmas, we end up with the following alternative formulations

$$\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}] \approx \boldsymbol{\mu}_* + \Phi_*(\Phi^\mathsf{T}\Phi + \sigma^2\Lambda^{-1})^{-1}\Phi^\mathsf{T}(\mathbf{y} - \boldsymbol{\mu}), \tag{3.45a}$$
$$\mathrm{Cov}\left[\boldsymbol{f}_*|\boldsymbol{y} = \mathbf{y}\right] \approx \sigma_n^2\Phi_*(\Phi^\mathsf{T}\Phi + \sigma^2\Lambda^{-1})^{-1}\Phi_*^\mathsf{T}. \tag{3.45b}$$

Instead of inverting a matrix of size $N \times N$, we now invert a matrix of size $m \times m$. Equivalent reformulations are available when computing the marginal likelihood; since the hyperparameter selection is the bottleneck of GP regression in terms of computation time, the overall time scaling is changed from $\mathcal{O}\left(N^3\right)$ to $\mathcal{O}\left(m^3\right)$ (similar efficient expressions are unfortunately not available for the cross validation approach). Moreover, the memory scaling is reduced from $\mathcal{O}\left(N^2\right)$ to $\mathcal{O}\left(mN\right)$.

Another advantage with this method is that the basis functions decouple the arguments $\mathbf{x}$ and $\mathbf{x}'$. This is particularly useful when considering linear functional evaluations: instead of computing $\mathcal{L}\mathcal{L}'k(\mathbf{x}, \mathbf{x}')$ we only need to compute $\mathcal{L}\phi_j(\mathbf{x})$, which is further simplified by the separate form of $\phi_j(\mathbf{x})$. For instance, this replaces tedious numerical integration with closed form expressions, a fact exploited in Paper II, III and V.

The number of basis functions m is limited by the computational resources, while L_k specifies the frequency resolution in direction k, as seen in Figure 3.2. An approach that has shown useful in practice is to select L_k so as to cover the vast amount of the spectral mass. Since the spectral density depends on the hyperparameters, this implies that the basis functions must be recomputed as the optimisation proceeds; however, this computational overhead can be limited to every q^{th} iteration. Also, note that updating the basis functions changes the approximation and therefore the definition of the cost function; this effect has small impact and is negligible in practise.

3.4 Deep kernel learning

Stationary covariance functions have a limited expressiveness, and performs poorly when modelling functions with discontinuous features. To overcome this, a wide range of non-stationary covariance functions have been developed; the most notable one being the neural network covariance function [22]. However, these covariance functions tend to be more challenging to work with and extend. For instance, they can not be approximated with the method described in Section 3.3.

Another way of increasing the expressiveness of a stationary covariance function is to transform the inputs through a nonlinear mapping $\mathbf{u}(\mathbf{x}) : \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}^{D_\mathbf{u}}$. The resulting covariance function $k(\mathbf{u}(\mathbf{x}), \mathbf{u}(\mathbf{x}'))$ is stationary with respect to $\mathbf{u}(\mathbf{x})$ and $\mathbf{u}(\mathbf{x}')$, but non-stationary with respect to the original inputs $\mathbf{x}$ and $\mathbf{x}'$. For instance, the mapping

$$\mathbf{u}(x) = \begin{bmatrix} \sin x \\ \cos x \end{bmatrix}, \tag{3.46}$$

is customised to model functions with periodic behaviour [7, 14, 27]. A highly expressive mapping able to describe general complex structures is obtained by letting $\mathbf{u}(\mathbf{x})$ be a neural network. This particular construction is referred to *manifold Gaussian processes* [5] or *deep kernel learning* [32].

The underlying intuition is that we use $\mathbf{u}(\mathbf{x})$ to separate the non-stationary regions of the input space, such that the corresponding function values are assigned low correlation in the stationary space. As an example, consider the modified sine function in the left plot of Figure 3.3, to which a sharp discontinuity is introduced at $x = 0$. The red dashed line illustrates the standard GP mapping $u(x) = x$. A sine function is well-modelled by a stationary covariance function, but a problem occurs close to the step; a small number $\epsilon > 0$ yields a small difference

$$u(\epsilon) - u(-\epsilon) = \epsilon - (-\epsilon) = 2\epsilon,$$

so $\not{f}(-\epsilon)$ and $\not{f}(\epsilon)$ are modelled as similar. The green dashed dotted line is a modified version of this mapping, in which the discontinuity has been accounted for; now $u(-\epsilon)$ and $u(\epsilon)$ are clearly different, which gives a lower covariance between $\not{f}(-\epsilon)$ and $\not{f}(\epsilon)$, as is desired. The right plot shows the predictions obtained with these mappings, given a set of measurements (red stars). It is seen that the slight modification of $u(x)$ yields a much better result, and it is for the detection of this kind of features that the neural network is used; the remaining parts are handled by the GP.

Furthermore, the example clearly illustrates the potential danger of misinterpreting the credible region. Although the standard GP prediction is very poor around the step, it still has a low variance; this overconfidence stems from the inaccurate stationarity assumptions.

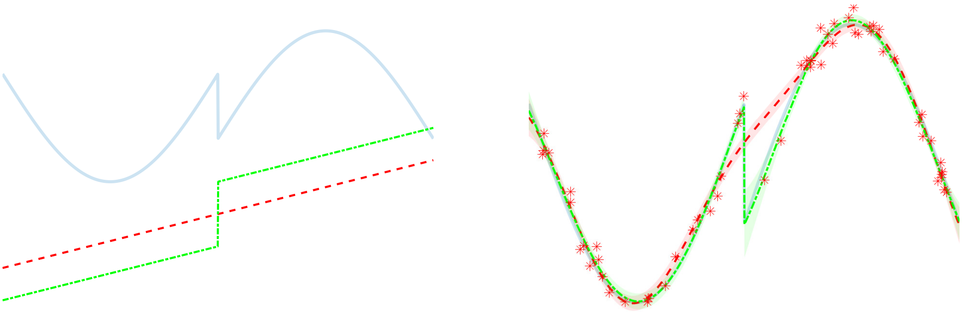


Figure 3.3: *Left*: target function and two different mappings $u(x)$: the first one $u(x) = x$ as in a standard GP (red dashed), while the other accounts for the discontinuity (green dashed dotted). *Right*: corresponding GP predictions.

Recall that $D_\mathbf{u}$ is the output dimension of $\mathbf{u}$, which in turn is the input provided to the GP. Furthermore, $D_\mathbf{u}$ can be chosen arbitrary, and is in no way constrained by the dimension $D_\mathbf{x}$ of $\mathbf{x}$. While a large $D_\mathbf{u}$ might provide more expressiveness, a small $D_{\mathbf{u}}$ is computationally more efficient. For instance, deep kernel learning can be used together with the approximation method described in Section 3.3, which is then applied in the $\mathbf{u}$-space. Since the number of basis functions scales exponentially with the input dimension, it is in that case computationally preferable to keep $D_{\mathbf{u}}$ low.

The hyperparameter selection is made the same way as previously described, with a slight modification in that the chain rule is required to compute the partial derivatives with respect to the parameters of $\mathbf{u}(\mathbf{x})$. However, the significant number of hyperparameters introduced by the neural network makes the optimisation problem much harder. The number of local optima is large, and the risk of ending up with a useless prediction is overwhelming. Hence, the initialisation is a crucial factor. Furthermore, we are faced with an intricate design problem; how should the network be chosen with respect to the number of neurons, the number of layers and choice of activation functions? The network must be large enough to provide the desired expressiveness, and small enough to limit the risk of over-fitting. Although these issues are well-known within deep learning research, they are nevertheless challenging.

Intuitively, the meeting between GPs and deep learning is appealing. However, having learnt to appreciate the power and potential of GP regression, it is natural and perfectly valid to feel some scepticism. The GP is, after all, a popular and successful model in no small part due to its ability to describe complex functions with a *small* number of free parameters. A heavily parameterised covariance function perturbs this fundamental property and reduces the robustness. However, the construction is still young and further developments are to be expected.

In Paper IV, we describe how deep kernel learning is applied to problems with integral measurements, using dimensionality reduction, the approximation method of Section 3.3 and a customised hyperparameter initialisation.

Chapter 4

Tomographic reconstruction

Tomography refers the procedure in which some form of penetrating waves are being sent through an object to produce lower dimensional projections of it; tomographic reconstruction refers to the process of describing the interior of the object given measurements generated in that procedure.

The most well-known tomographic application is X-ray imaging. Widely employed within medicine, this is a technique of which many people have practical experience. Due to that fact, it is perhaps easy to forget the non-trivial challenge of transforming the measured raw data to the images analysed by the doctor.

In this chapter we consider two types of tomographic problems: X-ray computed tomography (CT) and strain field reconstruction. The latter problem has gained a rising interest during recent years; it aims at quantifying deformation, for which the physical description is given by the *strain field*. While CT concerns inference of a scalar target function, the strain field is a tensor field and hence a multivariate quantity.

The tomographic problem can be intuitively understood by considering a generalisation of the so called *magic square*. A magic square of size $n \times n$ is built up by n^2 sub-squares, in which the integers $1, \ldots, n$ are distributed such that the sums along rows, columns and diagonals are equal; see the left of Figure 4.1 for the case $n = 3$, in which the solution is unique up to rotation and reflection.

Now consider a square containing any set of numbers, where we also take into account the sums along the sub-diagonals, as seen in the middle of Figure 4.1. Here, the numbers are given and the sums are easy to verify. In the right part of the figure, the sums have different values, and the numbers in the sub-squares are replaced by question marks. This constitutes a basic demonstration of the tomographic problem: *replace the question marks with numbers that obey the given sums.*

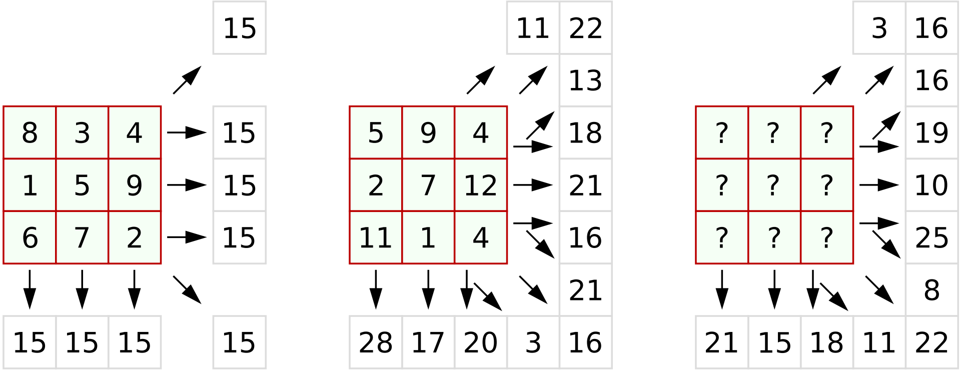


Figure 4.1: Illustration of the tomographic problem. *Left*: a magic square, where the sum is constant along rows, columns and diagonals. *Middle*: a square with arbitrary numbers and summations along different paths. *Right*: Given the summations, the problem is to fill in the missing numbers.

In the generalised case, the square is replaced by an irregularly shaped object, the numbers are allowed to be decimals and the sums are contaminated by measurement noise. Furthermore, the sub-squares are made infinitely small which implies that the sums contain infinitely many terms – in the limit, the sub-squares are replaced by points and the sums by integrals.

Of course, one can build a similar analogy in three dimensions, simply by replacing the sub-squares with cubes. Although the extension is conceptually straightforward, in this thesis we restrict our interest to two dimensions; if not for other reasons, it keeps the mathematical notation cleaner.

4.1 X-ray computed tomography

The goal in X-ray CT is to reconstruct the *attenuation function* $\mathtt{f}(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}$, that describes the interior of the investigated object. Basically, the experimental procedure consists of generating X-rays at a source, sending them through the object and recording them at a detector. The location of the source and the detector constitutes the *projection geometry*, where a projection refers to a set of rays generated with a similar setup. The most basic version is the *parallel beam geometry*, where the rays are propagating side-by-side in the same direction. This geometry is employed in the first generation CT scanners. As the source and detector has to be re-positioned for every single measurements, the technique is inefficient and no longer used in practise. However, the geometry is important for algorithm development and theoretical analysis. Later generations employ the *fan beam geometry*, where the source generates multiple rays simultaneously. This allows for the use of several detectors and thereby a faster data collecting process. Principal sketches of these two geometries are seen in Figure 4.2. In the latest, seventh generation CT scanners, rays are generated in the shape of a three-dimensional cone, which provides further benefits when aiming at reconstructing an entire three-dimensional volume.

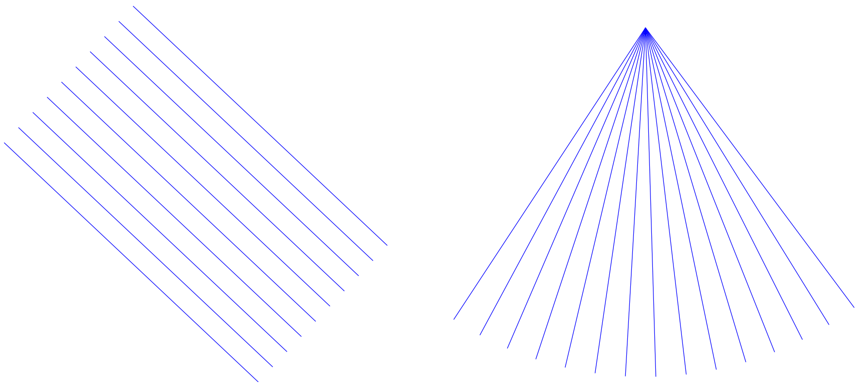


Figure 4.2: Illustration of parallel beam geometry (left) and fan beam geometry (right). Since the latter allows for generation of multiple X-rays simultaneously, it is more efficient in practise.

The attenuation function $\mathtt{f}(\mathbf{x})$ determines how the intensity of the X-rays are affected when travelling through the object. We use I_0 to denote the initial intensity of an X-ray at the source, while I denotes the measured intensity at the detector. The relation between I and I_0 is modelled through an exponential decay

$$I = I_0 \exp\left[-\int_{-R}^{R} \mathtt{f}(\mathbf{x}^0 + s\hat{\mathbf{n}})\, ds\right], \tag{4.1}$$

where $\mathbf{x}^0$ denotes the centre point of the ray and $\hat{\mathbf{n}}$ is a unit vector defining the direction of propagation. The integral is required since $\mathtt{f}(\mathbf{x})$ is assumed to be non-constant. Ideally, the integration limits $\pm R$ correspond to the object boundary; however, the position of the boundary is unknown, and instead we use the positions of the source and the detector. Rewriting (4.1) as

$$\log \frac{I_0}{I} = \int_{-R}^{R} \mathtt{f}(\mathbf{x}^0 + s\hat{\mathbf{n}})\, ds, \tag{4.2}$$

we obtain the core formulation of the CT problem. Assuming additive Gaussian noise, we can express the measurements as

$$y_i = \int_{-R}^{R} \mathtt{f}(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i)\, ds + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}\left(0,\ \sigma^2\right), \tag{4.3}$$

a notation recognised from the previous chapter. Indeed, this measurement model allows for an application of the GP; this is the focus of Paper III, which also provides neat theoretical insights by interpreting traditional CT algorithms in the GP context.

4.1.1 Filtered backprojection

Filtered backprojection (FBP) [17] is a very important algorithm for CT reconstruction, and has served as state-of-the-art for decades; it is fast, memory-efficient and fairly straightforward to derive in theory and implement in practice. Since it is used for comparison in Papers III and IV without much description, it is fair to devote some space for the basic derivation; more elaborate descriptions are found for instance in [4, 8]. Although the choice of projection geometry does not require an extended formulation of the GP framework, it does for FBP; here we restrict our interest to the parallel beam geometry and note that the more advanced alternatives follow along similar lines, but require more rigorous mathematical treatment. Furthermore, to facilitate readability we focus on continuous quantities, and then conclude the section with a discussion on some practical aspects.

To begin with, we introduce the *Radon transform*

$$g(\rho, \theta) = \iint_{\mathbb{R}^2} \mathtt{f}(\mathbf{x}) \delta(\hat{\boldsymbol{\rho}}^\mathsf{T} \mathbf{x} - \rho) \, d\mathbf{x}, \tag{4.4}$$

where

$$\hat{\boldsymbol{\rho}} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \tag{4.5}$$

and the delta function restricts the integration to the line $l(\rho, \theta)$, which consists of all points fulfilling

$$\hat{\boldsymbol{\rho}}^\mathsf{T} \mathbf{x} = \rho \quad \Leftrightarrow \quad x_1 \cos\theta + x_2 \sin\theta = \rho. \tag{4.6}$$

See Figure 4.3 for details. Hence, (4.4) is an alternative way of expressing the integral in (4.1); instead of using a unit vector and a centre point, the line is defined by the projection angle θ and the perpendicular distance to the origin ρ. Also note that integrating over the entire line does not change the value of the integral since there are no contributions from the region outside the object.

The image obtained when illustrating $g(\rho, \theta)$ in the $\{\rho, \theta\}$-plane is called a *sinogram*, and the process of reconstructing $\mathtt{f}(\mathbf{x})$ from its sinogram is referred to as *backprojection*. Up to a scale factor, the naive approach of backprojection is to directly map a point in the sinogram to the corresponding

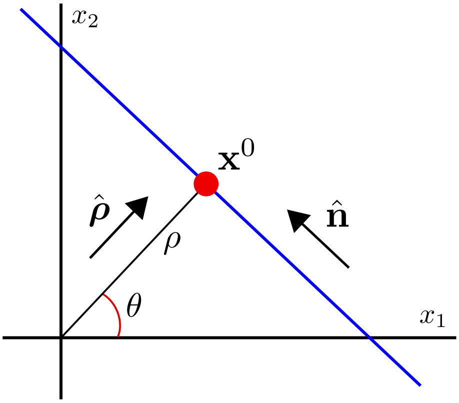


Figure 4.3: A line can be represented by a point and a unit vector, but also by an angle and the perpendicular distance to the origin.

line in $\mathbf{x}$-space; all points in $l(\rho, \theta)$ are assigned the value $g(\rho, \theta)$, and the estimate $\hat{\mathtt{f}}(\mathbf{x})$ of $\mathtt{f}(\mathbf{x})$ is obtained by integrating over all angles

$$\hat{\mathtt{f}}(\mathbf{x}) = \int_0^\pi g(\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}, \theta)\, d\theta. \tag{4.7}$$

The image resulting from this procedure is called a *laminogram*. The laminogram is inevitably blurry and misses out on sharp details, which follows from an incorrect treatment of the relation between $\mathtt{f}(\mathbf{x})$ and $g(\rho, \theta)$, as is clarified below.

We proceed our derivation by considering the one-dimensional Fourier transform of $g(\rho, \theta)$ computed with respect to ρ, which is given by

$$G(\xi, \theta) = \mathfrak{F}[g(\rho, \theta)] = \int_\mathbb{R} g(\rho, \theta) e^{-i2\pi\xi\rho}\, d\rho, \tag{4.8}$$

where ξ is the frequency variable. Substituting $g(\rho, \theta)$ with the expression in (4.4) yields

$$\begin{aligned} G(\xi, \theta) &= \int_\mathbb{R} \left[\iint_{\mathbb{R}^2} \mathtt{f}(\mathbf{x}) \delta(\hat{\boldsymbol{\rho}}\mathsf{T}\mathbf{x} - \rho)\, d\mathbf{x} \right] e^{-i2\pi\xi\rho}\, d\rho \\ &= \iint_{\mathbb{R}^2} \mathtt{f}(\mathbf{x}) \left[\int_\mathbb{R} \delta(\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x} - \rho) e^{-i2\pi\xi\rho}\, d\rho \right] d\mathbf{x} \\ &= \iint_{\mathbb{R}^2} \mathtt{f}(\mathbf{x}) e^{-i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}}\, d\mathbf{x}, \end{aligned} \tag{4.9}$$

where we have used the sifting property of the delta function

$$\int_\mathbb{R} \delta(x_0 - x) q(x)\, dx = q(x_0). \tag{4.10}$$

Introducing

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{4.11}$$

we can rewrite (4.9) as

$$\begin{aligned} G(\xi, \theta) &= \iint_{\mathbb{R}^2} \mathtt{f}(\mathbf{x}) e^{-i2\pi \mathbf{u}^\mathsf{T} \mathbf{x}} \, d\mathbf{x} \Big|_{\mathbf{u}=\xi\hat{\boldsymbol{\rho}}} \\ &= \mathfrak{F}[\mathtt{f}(\mathbf{x})](\mathbf{u}) \Big|_{\mathbf{u}=\xi\hat{\boldsymbol{\rho}}} \\ &= \mathfrak{F}[\mathtt{f}(\mathbf{x})](\xi\hat{\boldsymbol{\rho}}). \end{aligned} \tag{4.12}$$

Hence, $G(\xi, \theta)$ is the two-dimensional Fourier transform of $\mathtt{f}(\mathbf{x})$ evaluated at the line defined by

$$u_1 = \xi \cos\theta, \qquad u_2 = \xi \sin\theta. \tag{4.13}$$

This relation is known as the *Fourier-slice theorem*, and implies that a projection taken from a fixed angle in $\mathbf{x}$-space corresponds to a slice of the frequency space, see the top row of Figure 4.4. From this insight, we realise why the naive backprojection approach described above performs poorly; even when adding a large set of projections, low frequencies are clearly over-represented, as illustrated in the bottom row of Figure 4.4. Unless accounting for this unbalance, the reconstructed image becomes blurry since detailed shapes are described in the under-represented high frequencies.

The correct approach is instead as follows. First we note that $\mathtt{f}(\mathbf{x})$ is obtained as the inverse Fourier transform of $\mathfrak{F}[\mathtt{f}(\mathbf{x})](\mathbf{u})$, namely

$$\mathtt{f}(\mathbf{x}) = \mathfrak{F}^{-1}\left[\mathfrak{F}[\mathtt{f}(\mathbf{x})](\mathbf{u})\right](\mathbf{x}) = \iint_{\mathbb{R}^2} \mathfrak{F}[\mathtt{f}(\mathbf{x})](\mathbf{u}) e^{i2\pi \mathbf{u}^\mathsf{T} \mathbf{x}} \, d\mathbf{u}. \tag{4.14}$$

By making a change of variables according to the previous substitution (4.13), the integral becomes

$$\begin{aligned} \mathtt{f}(\mathbf{x}) &= \int_0^{2\pi} \int_0^{\infty} \mathfrak{F}[\mathtt{f}(\mathbf{x})](\xi\hat{\boldsymbol{\rho}}) e^{i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \xi \, d\xi \, d\theta \\ &= \int_0^{2\pi} \int_0^{\infty} G(\xi, \theta) e^{i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \xi \, d\xi \, d\theta, \end{aligned} \tag{4.15}$$

where the last equality follows from the Fourier slice theorem (4.12). Furthermore, since

$$\begin{bmatrix} \cos(\theta + \pi) \\ \sin(\theta + \pi) \end{bmatrix} = -\begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} = -\hat{\boldsymbol{\rho}}, \tag{4.16}$$

it follows that

$$G(\xi, \theta + \pi) = \mathfrak{F}[\mathtt{f}(\mathbf{x})](-\xi\hat{\boldsymbol{\rho}}) = G(-\xi, \theta). \tag{4.17}$$

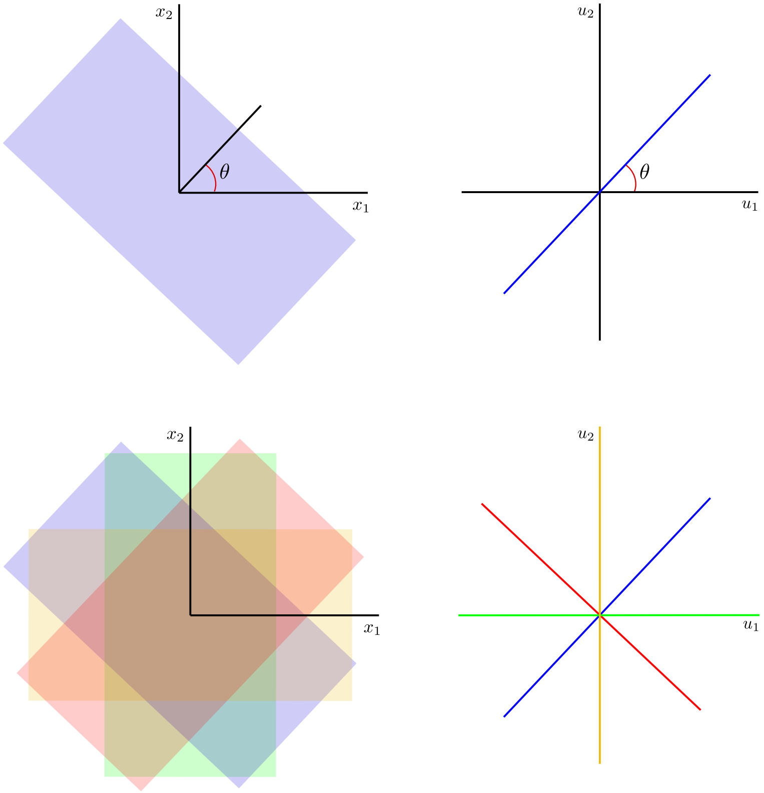


Figure 4.4: Illustration of the Fourier-slice theorem. *Top row*: A projection in **x**-space corresponds to a slice of the frequency space. *Bottom row*: As several projections are considered, we see that the frequency representation is unbalanced, with low frequencies being over-represented.

The close connection is intuitively understood as projections taken from the angles θ and $\theta + \pi$ contain the same information, since they simple correspond to exchanging the positions of the source and the detector. Now note that the integral in (4.15) can be split into two parts

$$\begin{aligned} \mathtt{f}(\mathbf{x}) = & \underbrace{\int_0^{\pi} \int_0^{\infty} G(\xi, \theta) e^{i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \xi \, d\xi \, d\theta}_{I_1} \\ & + \underbrace{\int_{\pi}^{2\pi} \int_0^{\infty} G(\xi, \theta) e^{i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \xi \, d\xi \, d\theta,}_{I_2} \end{aligned} \tag{4.18}$$

and we use (4.16)-(4.17) to reformulate the second integral as

$$\begin{aligned} I_2 &= \int_0^\pi \int_0^\infty G(\xi, \theta + \pi) e^{-i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \xi \, d\xi \, d\theta \\ &= \int_0^\pi \int_0^\infty G(-\xi, \theta) e^{-i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \xi \, d\xi \, d\theta \\ &= \int_0^\pi \int_{-\infty}^0 G(\xi, \theta) e^{i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} |\xi| \, d\xi \, d\theta. \end{aligned} \tag{4.19}$$

Finally, we regain a single integral as

$$\mathtt{f}(\mathbf{x}) = I_1 + I_2 = \int_0^\pi \int_{-\infty}^\infty |\xi| G(\xi, \theta) e^{i2\pi\xi\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x}} \, d\xi \, d\theta. \tag{4.20}$$

The inner integral over ξ constitutes a *filtered* one-dimensional inverse Fourier transform of $G(\xi, \theta)$; we refer to $|\xi|$ as a *Ram-Lak* filter or *ramp* filter. This high-pass filter compensates for the unbalanced frequency representation we discussed earlier, and explains the name *filtered* backprojection.

There is a theoretical problem in that the filter causes the integral to diverge. Hence, it is common practise to use a window $W(\xi)$ that cancels all frequencies outside a certain range; we thus replace $|\xi|$ with $S(\xi) = |\xi| W(\xi)$. As we have a multiplication $S(\xi)G(\xi, \theta)$ in the Fourier domain, the inner integral can be equivalently expressed directly in the spatial domain as a convolution between the space functions $s(\rho)$ and $g(\rho, \theta)$, according to

$$\mathtt{f}(\mathbf{x}) = \int_0^\pi \int_{-\infty}^\infty s(\hat{\boldsymbol{\rho}}^\mathsf{T}\mathbf{x} - \rho) g(\rho, \theta) \, d\rho \, d\theta. \tag{4.21}$$

For optimised computational efficiency, this formulation tend to be preferred in practise [8].

A few more notes should be made about the practical implementation. First of all, our measurements corresponds to a finite set of sinogram samples $\{g(\rho_i, \theta_i)\}_{i=1}^N$, and so the continuous integrals above are replaced by discrete sums to form the estimate of $\mathtt{f}(\mathbf{x})$. Note that due to this discrete nature, interpolation is required to form the reconstructed image, and hence the estimate in an arbitrary test point $\mathbf{x}_*$ is not directly provided. Unsurprisingly, the sampling is crucial; the FBP reconstruction is known to introduce streaking artefacts due to aliasing effects, and a large set of projection angles is necessary to obtain a satisfying result. Moreover, the algorithm makes no attempt of accounting for measurement noise and is indeed quite sensitive to noise. Also, as previously mentioned, FBP is geometry-dependent; the mathematics of the fan beam geometry is not as straightforward to derive, and the extension to three dimensions is even more complex.

From a computationally point of view, however, FBP demonstrates an outstanding performance in time and memory. The final reconstruction can be updated sequentially, since it is the sum of the individual reconstructions from each projection; these are in turn efficiently computed as discrete convolutions, or alternatively in the frequency domain using the fast Fourier transform. Furthermore, the algorithm has few free parameters – basically, the only user choice required is the window function.

4.1.2 Beyond filtered back projection

The FBP algorithm is a so called *direct method*: all computations are made once, in a single step. In contrast, *iterative methods* are constituted by updates in several steps, typically through an optimisation routine. The interest in these methods is high, and recent developments show an impressive potential; in particular, several deep learning based constructions outperform FBP in image quality when the data sets are small [1, 9, 24]. However, the superior reconstruction performance has a price in complex methodologies and time-consuming, expensive computations. Partially for this reason, FBP is still the dominating reconstruction algorithm in practise, and it may be hard to see any reason not to keep it that way; since data has to be collected, may we not as well collect a large amount of it?

The answer is: not necessarily. In fact, there is a particular interest not just in the *limited data problem*, but also in the *limited angle problem*. The are several reasons for this. X-ray scans are expensive to operate, and efficient algorithms can speed up the throughput as less data is required. Also, exposing the human body for high radiation doses can potentially harm the tissue, and keeping the doses small reduces this risk. Furthermore, the experimental setup does not necessarily allow for projections taken from certain angles, as is the case for instance in mammography. Moreover, the time aspect is important to consider for another reason; moving objects, such as a beating heart, requires a fast scan which limits the amount of data collected.

It is important to recall that established frameworks takes time to change – this requires a close connection between the theorists and CT engineers, and new algorithms are not put into place as easy and fast as one would hope. Even with source code made freely available by researchers, the implementation may often require customised modifications hard to achieve unless you actually understand the underlying theory and is familiar with the software requirements. At the same time, other practical issues are competing with the reconstruction algorithm about the engineers' attention, including hardware developments aimed at reducing the noise present during the scanning procedure [23].

Nevertheless, eventually a new generation of reconstruction methods will replace FBP as the state-of-the-art not just in theory, but also in practise. The computational burdens are more and more reduced, in no small part through efficient parallel computations enabled by powerful graphical processing units.

4.1.3 Deep kernel learning for CT reconstruction

As mentioned in the introduction of this chapter, Paper III concerns GP reconstruction for CT imaging. Although promising results are shown in comparison with FBP in the limited data setting, the model is still struggling and can not compete with more sophisticated developments in the literature. The explanation for this is that CT images tend to contain discontinuities; for instance, the human body has sharp transitions between flesh and bones. We have previously discussed how such *non-stationary* features make the GP regression challenging. In Section 3.4, we reviewed the concept of deep kernel learning, where the inputs to a standard covariance function are warped through a neural network in order to increase the expressiveness. In Paper IV, we describe a practical procedure of how to apply deep kernel learning for integral measurement, illustrated by CT examples. The reconstruction performance is significantly improved and indicate a potential for further research in that field.

4.2 Strain field reconstruction

The second tomographic problem we consider is reconstruction of elastic strain fields within deformed polycrystalline materials, where our goal is to reconstruct the strain tensor

$$\boldsymbol{\epsilon}(\mathbf{x}) = \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) & \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{yx}(\mathbf{x}) & \epsilon_{yy}(\mathbf{x}) \end{bmatrix} \in \mathbb{R}^{2\times 2}. \tag{4.22}$$

This tensor is symmetric, leaving us with three unique components required to quantify the deformation in a two-dimensional object, see Figure 4.5.

The data is generated from *Bragg-edge analysis*, which rely on neutron beams as opposed to the previously discussed X-rays. Advanced high resolution time-of-flight neutron detectors are used to measure the arrival time of the neutrons, and the corresponding wavelength is determined through its direct relation with the velocity. Furthermore, the relative transmission rate is recorded, which is the ratio of the intensities measured without and with the sample present.

Within the material, there are a large number of randomly oriented crystal lattice planes, and the distance between them is called the *lattice spacing*.

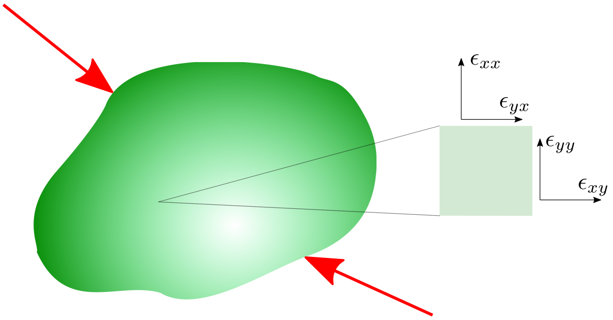


Figure 4.5: When an object is exposed to external forces, it is being deformed. The deformation is described by the elements of the strain tensor, which is symmetric so $\epsilon_{yx} = \epsilon_{xy}$. Hence, three components are required to quantify the strain at any point in a two-dimensional object.

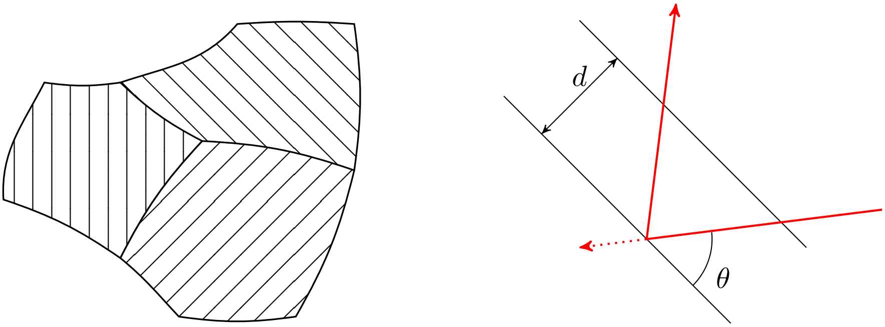


Figure 4.6: *Left*: Illustration of a crystal structure, consisting of crystals planes oriented in different angles. *Right*: Neutrons passing through the material interact with the crystal planes and are diffracted according to Bragg's law (4.23).

The neutrons interact with the crystal lattice planes, resulting in constructive diffraction modelled by *Bragg's law*

$$\lambda = 2d \sin \theta. \tag{4.23}$$

Here λ is the neutron wavelength, d is the lattice spacing and θ is the scattering angle, see Figure 4.6. From Bragg's law, we see that a larger wavelength results in a larger scattering angle. When $\theta = 90°$, the neutrons are scattered back to the source; this is referred to as *backscattering*. For larger wavelengths, constructive diffraction can not occur and the neutrons continue straight through the material. The result is a sharp, sudden increase of the intensity as a function of wavelength – this is called a *Bragg-edge*. The material contains several lattice spacings, which hence give rise to multiple Bragg-edges.

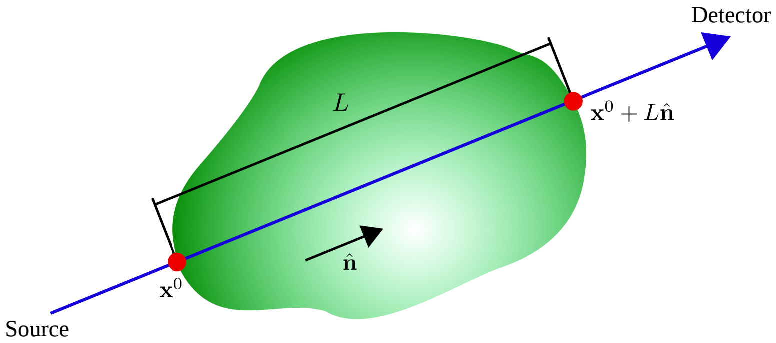


Figure 4.7: Illustration of the experimental setup. Neutrons beams generated at the source enter the object at the point $\mathbf{x}^0$, leave at $\mathbf{x}^0 + \hat{\mathbf{n}}L$ and are recorded by the detector.

The position of a Bragg-edge is used to determine the corresponding lattice spacing. The experiment is performed before and after the object is deformed, resulting in the different lattice spacings d_0 and d, respectively. From these, the average strain along the direction of propagation is obtained as

$$\langle \epsilon \rangle = \frac{d - d_0}{d_0}. \tag{4.24}$$

In terms of the strain tensor, this result is expressed with the *longitudinal ray transform* (LRT) [19]

$$\langle \epsilon \rangle = \frac{1}{L} \int_0^L \hat{\mathbf{n}}^\mathsf{T} \boldsymbol{\epsilon}(\mathbf{x}^0 + s\hat{\mathbf{n}})\hat{\mathbf{n}}\, ds. \tag{4.25}$$

Here, $\mathbf{x}^0$ denotes the point where the neutrons enter the object, the unit vector $\hat{\mathbf{n}}$ specifies their direction, and L is the total length of the sample along this line, see Figure 4.7. Note that the normalisation with L requires knowledge of the entry and exit points, although the determination of their positions is beyond the scope of this thesis; for a more elaborate discussion on the analysis of Bragg-edges, see for instance [28].

In order to fit the formulation into the GP framework, we use the symmetry of the strain tensor and rewrite the integrand as $\vec{\mathbf{n}}^\mathsf{T}\mathtt{f}(\mathbf{x}^0 + s\hat{\mathbf{n}})$, where

$$\mathtt{F}(\mathbf{x}) = \begin{bmatrix} f_{xx}(\mathbf{x}) \\ f_{xy}(\mathbf{x}) \\ f_{yy}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) \\ \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{yy}(\mathbf{x}) \end{bmatrix}, \qquad \text{and} \qquad \vec{\mathbf{n}} = \begin{bmatrix} n_x^2 \\ 2n_x n_y \\ n_y^2 \end{bmatrix}. \tag{4.26}$$

We now model the measurements $\{y_i\}_{i=1}^N$ according to

$$y_i = \frac{1}{L_i} \int_0^{L_i} \vec{\mathbf{n}}_i^\mathsf{T} \mathcal{F}(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i)\, ds + \varepsilon_i, \tag{4.27}$$

where $\mathcal{F}(\mathbf{x})$ is the GP model of $\mathtt{F}(\mathbf{x})$. Paper II elaborates this problem and demonstrates the GPs performance on both synthetic and real data set. This includes the construction of an invariant covariance function customised through exploitation of physical knowledge (see Section 3.2.3). In particular, restricting the design to the scalar potential called the *Airy stress function* explicitly guarantees fulfilment of the essential *equilibrium constraints*.

Although the framework introduced above allows for a wide range of applications, it excludes situations in which the undeformed lattice spacing is spatially dependent $d_0(\mathbf{x})$. Since this may occur as the result of several manufacturing techniques, a theoretical treatment is of practical interest. Under such circumstances, a naive approach assuming constant lattice spacing is doomed to fail. An extension of the LRT to the more general case is given by

$$\frac{1}{L}\int_0^L \left[\hat{\mathbf{n}}^\mathsf{T}\mathtt{F}(\mathbf{x}^0 + s\hat{\mathbf{n}}) + 1\right] d_0(\mathbf{x}^0 + s\hat{\mathbf{n}})\, ds. \tag{4.28}$$

A natural idea is to model both $\mathtt{F}(\mathbf{x})$ and $d_0(\mathbf{x})$ in a joint GP. However, a challenge is faced in that this generalisation constitutes a nonlinear functional, which breaks the Gaussianity. Hence, the GP regression can no longer be performed in closed form, and *approximative inference* is required. In Paper V, a variational inference technique is used to this end. Furthermore, the paper demonstrates how boundary conditions are exploited to ensure convergence.

Chapter 5

Concluding remarks

This chapter briefly summarises the thesis and discuss potential topics of future work.

5.1 Conclusions

The contribution of this thesis consists in practical approaches for advanced GP modelling, with a particular focus on tomographic applications.

We show how linear constraints can be embedded within the GP model, which accounts for physical knowledge and improves the regression performance. Along these lines we make a tailored design for strain field reconstruction, where the essential equilibrium constraints are explicitly incorporated. We also present an extension to strain fields characterised by nonlinear behaviour.

Furthermore, we use the GP for X-ray CT reconstruction, through which an alternative interpretation of established algorithms is obtained. Moreover, we use the CT problem to demonstrate the potential of deep kernel learning for integral measurements, where a neural network is incorporated in a standard covariance function.

5.2 Future work

The work included in this thesis opens up for a number of potential extensions and improvements, some of which that are already being addressed.

The linearly constrained GP relates the target function to an underlying potential function through a nonlinear transformation. While our approach of constructing this transformation has a practical focus, it is intriguing to ask how its mathematical properties can be described in general? A solid answer to this question is given in [18]. Another natural extension is to consider the more general case of nonlinear constraints. This problem might be attacked in a similar way using approximative inference, or alternatively through local linear approximations. A seemingly less challenging task – but perhaps more important – is to identify meaningful real-world problems that can benefit from such developments.

The tomographic reconstruction problems considered focus on the two-dimensional setting, whereas the three-dimensional case is more interesting in practice. For X-ray CT, the GP extension is trivial in theory, considering that the target function remains scalar. The strain problem is more complex, since the three-dimensional tensor field has six unique components; this in turn requires a more sophisticated covariance model. Promising work in this direction is presented in [10, 12], and is likely to be followed by further developments.

Having shown that deep kernel learning is a promising tool for CT problems, an interesting idea is to use it for strain reconstruction as well. The theoretical formulation is indeed straightforward; however, since the strain components are mapped from an underlying stress function, the implementation is more complicated. Also, more research is required for increased robustness and improved practical efficiency.

Appendix A

Solving the eigenvalue problem

The problem is to solve

$$\begin{cases} -\Delta\phi_j(\mathbf{x}) = \lambda_j\phi_j(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \phi_j(\mathbf{x}) = 0, & \mathbf{x} \in \partial\Omega, \end{cases} \tag{A.1}$$

where $\Omega = [-L_1, L_1] \times \cdots \times [-L_{D_\mathbf{x}}, L_{D_\mathbf{x}}]$. Using separation of variables, we seek solutions on the form

$$\phi_j(\mathbf{x}) = \prod_{k=1}^{D_\mathbf{x}} \varphi_{j_k}(x_k). \tag{A.2}$$

Plugging this into (A.1) and using the definition of the Laplace operator, we get

$$-\sum_{q=1}^{D_\mathbf{x}} \frac{\partial^2}{\partial x_q^2} \prod_{k=1}^{D_\mathbf{x}} \varphi_{j_k}(x_k) = \lambda_j \prod_{k=1}^{D_\mathbf{x}} \varphi_{j_k}(x_k). \tag{A.3}$$

Dividing both sides with the product yields

$$-\sum_{k=1}^{D_\mathbf{x}} \frac{1}{\varphi_{j_k}(x_k)} \frac{\partial^2}{\partial x_k^2} \varphi_{j_k}(x_k) = \lambda_j. \tag{A.4}$$

Since the right hand side is constant, the independent terms on the left hand side must also be constant. Furthermore, we require that $\varphi_{j_k}(x_k)$ is bounded for all x_k, and hence we have

$$-\frac{1}{\varphi_{j_k}(x_k)} \frac{\partial^2}{\partial x_k^2} \varphi_{j_k}(x_k) = c_{j_k}^2, \tag{A.5}$$

with the trigonometric solution

$$\varphi_{j_k}(x_k) = A_{j_k}\cos(c_{j_k}x_k) + B_{j_k}\sin(c_{j_k}x_k). \tag{A.6}$$

The boundary condition

$$\varphi_{j_k}(L_k) = \varphi_{j_k}(-L_k) = 0, \tag{A.7}$$

requires that $\varphi_{j_k}(L_k)$ is one of the functions

$$A_1\cos\left(\frac{\pi}{2L_k}x_k\right),\ B_2\sin\left(2\frac{\pi}{2L_k}x_k\right),\ A_3\cos\left(3\frac{\pi}{2L_k}x_k\right),\ \dots \tag{A.8}$$

See Figure A.1 for an illustration. This implies that

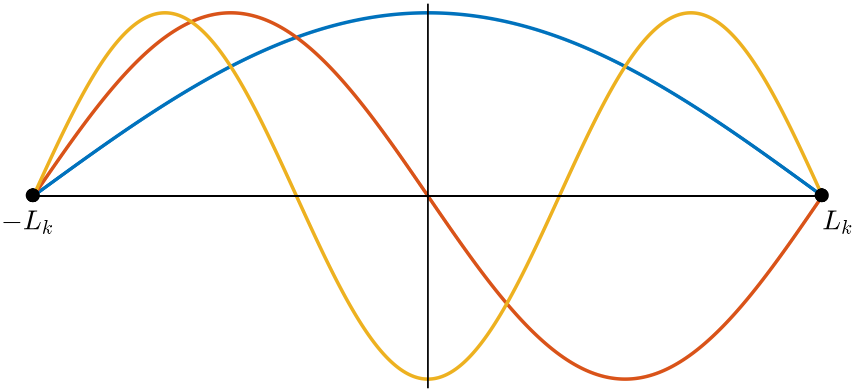


Figure A.1: The first three trigonometric functions fulfilling the boundary condition.

$$c_{j_k} = \frac{j_k\pi}{2L_i}, \qquad j_k \in \{1,2,3,\dots\}, \tag{A.9}$$

$$A_{j_k} = 0 \text{ for even } j_k, \quad B_{j_k} = 0 \text{ for odd } j_k. \tag{A.10}$$

Moreover, we want the basis functions to be orthonormal

$$1 = \int_{-L_k}^{L_k} \varphi_{j_k}(x_k)\varphi_{j_k}(x_k)\,dx_k = \begin{cases} A_{j_k}^2 L_k, & \text{for even } j_k, \\ B_{j_k}^2 L_k, & \text{for odd } j_k, \end{cases} \tag{A.11}$$

which in combination with (A.10) gives

$$A_{j_k} = L_k^{-1/2}\sin(c_{j_k}L_k), \tag{A.12a}$$

$$B_{j_k} = L_k^{-1/2}\cos(c_{j_k}L_k). \tag{A.12b}$$

Hence, we have that

$$\begin{aligned}\varphi_{j_k}(x_k) &= L_k^{-1/2}[\sin(c_{j_k}L_k)\cos(c_{j_k}x_k) + \cos(c_{j_k}L_k)\sin(c_{j_k}x_k)] \\ &= L_k^{-1/2}\sin\left(c_{j_k}(x_k + L_k)\right).\end{aligned} \tag{A.13}$$

Collecting all k contributions, we end up with

$$\phi_j(\mathbf{x}) = \prod_{k=1}^{D_{\mathbf{x}}} L_k^{-1/2}\sin\left(c_{j_k}(x_k + L_k)\right), \tag{A.14}$$

where from (A.4)-(A.5) we have

$$\lambda_j = \sum_{k=1}^{D_{\mathbf{x}}} c_{j_k}^2. \tag{A.15}$$

References

[1] J. Adler and O. Öktem. “Learned Primal-Dual Reconstruction”. In: *IEEE Transactions on Medical Imaging* 37.6 (2018), pp. 1322–1332.

[2] M. A. Álvarez, L. Rosasco and N. D. Lawrence. “Kernels for Vector-Valued Functions: A Review”. In: *Foundations and Trends in Machine Learning* 4.3 (Mar. 2012), pp. 195–266.

[3] L. Baldassarre, L. Rosasco, A. Barla and A. Verri. “Vector Field Learning via Spectral Filtering”. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by J. L. Balcázar, F. Bonchi, A. Gionis and M. Sebag. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 56–71.

[4] T. M. Buzug. *Computed Tomography: From Photon Statistics to Modern Cone-Beam CT*. Springer-Verlag Berlin Heidelberg, 2008.

[5] R. Calandra, J. Peters, C. E. Rasmussen and M. P. Deisenroth. “Manifold Gaussian Processes for Regression”. In: *In Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. 2016, pp. 3338–3345.

[6] R. Garnett. *Lecture 11: Bayesian Quadrature*. University Lecture. Accessed: 2019-05-01. Available from: `http://www.cse.wustl.edu/~garnett/cse515t/spring_2018/files/lecture_notes/11.pdf`. 2018.

[7] N. H. Ghassemi and M. P. Deisenroth. “Analytic Long-Term Forecasting with Periodic Gaussian Processes”. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2014.

[8] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. ISBN: 013168728X.

[9] K. Hammernik, T. Würfl, T. Pock and A. Maier. “A Deep Learning Architecture for Limited-Angle Computed Tomography Reconstruction”. In: *Bildverarbeitung für die Medizin 2017*. Springer Berlin Heidelberg, 2017, pp. 92–97.

[10] J. Hendriks, A. Gregg, R. Jackson, C. Wensrich, A. Wills, A. Tremsin, T. Shinohara, V. Luzin and O. Kirstein. *Tomographic Reconstruction of Triaxial Strain Fields from Bragg-Edge Neutron Imaging.* Tech. rep. arXiv:1906.08506, June 2019.

[11] J. N. Hendriks, C. Jidling, T. B. Schön, A. Wills, C. M. Wensrich and E. H. Kisi. "Neutron transmission strain tomography for non-constant stress-free lattice spacing". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 456 (2019), pp. 64–73.

[12] J. N. Hendriks, C. M. Wensrich and A. Wills. *A Bayesian Approach to Triaxial Strain Tomography from High-energy X-ray Diffraction.* Tech. rep. arXiv:1903.02158, Mar. 2019.

[13] P. Hennig and M. Kiefel. "Quasi-Newton methods: A new direction". In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 843–865.

[14] D. J. C. MacKay. "Introduction to Gaussian Processes". In: *NATO ASI Series F Computer and Systems Sciences* 168 (1998), pp. 133–166.

[15] C. Jidling, T. B. Schön, J. Hendriks and A. Wills. *Deep kernel learning for integral measurements.* Tech. rep. arXiv:1909.01844, Sept. 2019.

[16] C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. "Probabilistic modelling and reconstruction of strain". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

[17] A. C. Kak and M. Slaney. *Principles of computerized tomographic imaging.* IEEE press New York, 1988.

[18] M. Lange-Hegermann. "Algorithmic Linearly Constrained Gaussian Processes". In: *Advances in Neural Information Processing Systems (NeurIPS).* 2018, pp. 2137–2148.

[19] W. R. B. Lionheart and P. J. Withers. "Diffraction tomography of strain". In: *Inverse Problems* 31.4 (2015), p. 045005.

[20] I. Macêdo and R. Castro. *Learning divergence-free and curl-free vector fields with matrix-valued kernels.* Tech. rep. Instituto Nacional de Matematica Pura e Aplicada, 2008.

[21] R. von Mises. *Mathematical Theory of Probability and Statistics.* Academic Press, 1964.

[22] R. M. Neal. *Bayesian Learning for Neural Networks.* Vol. 118. Lecture Notes in Statistics. Springer New York, 1996. ISBN: 9780387947242.

[23] X. Pan, E. Y. Sidky and M. Vannier. “Why do commercial CT scanners still employ traditional, filtered back-projection for image reconstruction?” In: *Inverse Problems* 25.12 (2009), p. 123009.

[24] D. M. Pelt, K. J. Batenburg and J. A. Sethian. “Improving Tomographic Reconstruction from Limited Data Using Mixed-Scale Dense Convolutional Neural Networks”. In: *Journal of Imaging* 4.11 (2018). ISSN: 2313-433X.

[25] Z. Purisha, C. Jidling, N. Wahlström, T. Schön and S. Särkkä. “Probabilistic approach to limited-data computed tomography reconstruction”. In: *Inverse Problems* 35.10 (2019), p. 105004.

[26] J. Quiñonero Candela, C. E. Ramussen and C. K. I. Williams. “Approximation Methods for Gaussian Process Regression”. In: *Large-scale kernel machines* (Sept. 2007), pp. 203–223.

[27] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning.* MIT press, Cambridge, MA, 2006.

[28] J. R. Santisteban, L. Edwards, M. E. Fizpatrick, A. Steuwer and P. J. Withers. “Engineering applications of Bragg-edge neutron transmission”. In: *Applied Physics* 74.1 (2002), pp. 1433–1436.

[29] A. Solin, M. Kok, N. Wahlström, T. B. Schön and S. Särkkä. “Modeling and Interpolation of the Ambient Magnetic Field by Gaussian Processes”. In: *IEEE Transactions on Robotics* 34.4 (Aug. 2018), pp. 1112–1127.

[30] A. Solin and S. Särkkä. “Hilbert space methods for reduced-rank Gaussian process regression”. In: *Statistics and Computing* (Aug. 2019).

[31] N. Wahlström and M. Kok. “Modeling magnetic fields using Gaussian processes”. In: *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP).* 2013, pp. 3522–3526.

[32] A. G. Wilson, Z. Hu, R. Salakhutdinov and E. P. Xing. “Deep Kernel Learning”. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS).* Cadiz, Spain, May 2016, pp. 370–378.

Paper I

Title

Linearly constrained Gaussian processes.

Authors

Carl Jidling, Niklas Wahlström, Adrian Wills, and Thomas B. Schön.

Statement of contribution

The development of the approach presented in this paper involved all authors. Carl Jidling and Niklas Wahlström was equally responsible for the implementations and most of the writing, with important contributions and feedback provided by Adrian Wills and Thomas B. Schön.

Edited version of

C. Jidling, N. Wahlström, A. Wills and T. B. Schön. "Linearly constrained Gaussian processes". In: *Advances in Neural Information Processing Systems (NIPS).* Los Angeles, US, 2017, pp. 1215–1224.

Linearly constrained Gaussian processes

Abstract

We consider a modification of the covariance function in Gaussian processes to correctly account for known linear operator constraints. By modelling the target function as a transformation of an underlying function, the constraints are explicitly incorporated in the model such that they are guaranteed to be fulfilled by any sample drawn or prediction made. We also propose a constructive procedure for designing the transformation operator and illustrate the result on both simulated and real-data examples.

1 Introduction

Bayesian non-parametric modelling has had a profound impact in machine learning due, in no small part, to the flexibility of these model structures in combination with the ability to encode prior knowledge in a principled manner [7]. These properties have been exploited within the class of Bayesian non-parametric models known as Gaussian Processes (GPs), which have received significant research attention and have demonstrated utility across a very large range of real-world applications [18].

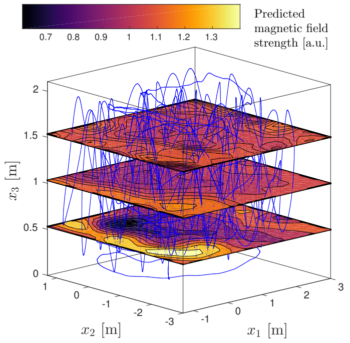


Figure 1: Predicted strength of a magnetic field at three heights, given measured data sampled from the trajectory shown (blue curve). The three components (x_1, x_2, x_3) denote the Cartesian coordinates, where the x_3-coordinate is the height above the floor. The magnetic field is curl-free, which can be formulated in terms of three linear constraints. The method proposed in this paper can exploit these constraints to improve the predictions. See Section 5.2 for details.

Abstracting from the myriad number of these applications, it has been observed that the efficacy of GPs modelling is often intimately dependent on the appropriate choice of mean and covariance functions, and the appropriate tuning of their associated hyperparameters. Often, the most appropriate mean and covariance functions are connected to prior knowledge of the underlying problem. For example, [11] uses functional expectation constraints to consider the problem of gene-disease association, and [14] employs a multivariate generalised von Mises distribution to produce a GP-like regression that handles circular variable problems.

At the same time, it is not always obvious how one might construct a GP model that obeys underlying principles, such as equilibrium conditions and conservation "laws". One straightforward approach to this problem is to add fictitious measurements that observe the constraints at a finite number of points of interest. This has the benefit of being relatively straightforward to implement, but has the sometimes significant drawback of increasing the problem dimension and at the same time not enforcing the constraints between the points of interest.

A different approach to constraining the GP model is to construct mean and covariance functions that obey the constraints. For example, curl and divergence free covariance functions are used in [24] to improve the accuracy for regression problems. The main benefit of this approach is that the problem dimension does not grow, and the constraints are enforced everywhere, not pointwise. However, it is not obvious how these approaches can be scaled for an arbitrary set of linear operator constraints. The contribution of this paper is a new way to include constraints into multivariate GPs. In particular, we develop a method that transforms a given GP into a new, derived, one that satisfies the constraints. The procedure relies upon the fact that GPs are closed under linear operators, and we propose an algorithm capable of constructing the required transformation. We will demonstrate the utility of this new method on both simulated examples and on a real-world application, the latter in form of predicting the components of a magnetic field, as illustrated in Figure 1.

To make these ideas more concrete, we present a simple example that will serve as a focal point several times throughout the paper. To that end, assume that we have a two-dimensional function $\mathbf{f}(\mathbf{x}) : \mathbb{R}^2 \mapsto \mathbb{R}^2$ on which we put a GP prior $\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}\left(\boldsymbol{\mu}(\mathbf{x}),\ \mathbf{K}(\mathbf{x}, \mathbf{x}')\right)$. We further know that $\mathbf{f}(\mathbf{x})$ should obey the differential equation

$$\frac{\partial f_1}{\partial x_1} + \frac{\partial f_2}{\partial x_2} = 0. \tag{1}$$

In this paper we show how to modify $\mathbf{K}(\mathbf{x}, \mathbf{x}')$ and $\boldsymbol{\mu}(\mathbf{x})$ such that any sample from the new GP is guaranteed to obey the constraints like (1), considering any kind of linear operator constraint.

2 Problem formulation

Assume that we are given a data set of N observations $\{\mathbf{x}_k, \mathbf{y}_k\}_{k=1}^N$ where $\mathbf{x}_k$ denotes the input and $\mathbf{y}_k$ the output. Both the input and output are potentially vector-valued, where $\mathbf{x}_k \in \mathbb{R}^D$ and $\mathbf{y}_k \in \mathbb{R}^K$. We consider the regression problem where the data can be described by a non-parametric model $\mathbf{y}_k = \mathbf{f}(\mathbf{x}_k) + \mathbf{e}_k$, where $\mathbf{e}_k$ is zero-mean white noise representing the measurement uncertainty. In this work, we place a vector-valued GP prior on $\mathbf{f}$

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}\left(\boldsymbol{\mu}(\mathbf{x}),\ \mathbf{K}(\mathbf{x}, \mathbf{x}')\right), \tag{2}$$

with the mean function and the covariance function

$$\boldsymbol{\mu}(\cdot):\ \mathbb{R}^D \mapsto \mathbb{R}^K, \qquad \mathbf{K}(\cdot,\cdot):\ \mathbb{R}^D \times \mathbb{R}^D \mapsto \mathbb{R}^K \times \mathbb{R}^K. \tag{3}$$

Based on the data $\{\mathbf{x}_k, \mathbf{y}_k\}_{k=1}^N$, we would now like to find a posterior over the function $\mathbf{f}(\mathbf{x})$. In addition to the data, we know that the function $\mathbf{f}$ should fulfil certain constraints

$$\mathscr{F}_{\mathbf{x}}[\mathbf{f}] = \mathbf{0}, \tag{4}$$

where $\mathscr{F}_{\mathbf{x}}$ is an operator mapping the function $\mathbf{f}(\mathbf{x})$ to another function $\mathbf{g}(\mathbf{x})$ as $\mathscr{F}_{\mathbf{x}}[\mathbf{f}] = \mathbf{g}(\mathbf{x})$. We further require $\mathscr{F}_{\mathbf{x}}$ to be a linear operator meaning that

$$\mathscr{F}_{\mathbf{x}}\Big[\lambda_1\mathbf{f}_1 + \lambda_2\mathbf{f}_2\Big] = \lambda_1\mathscr{F}_{\mathbf{x}}[\mathbf{f}_1] + \lambda_2\mathscr{F}_{\mathbf{x}}[\mathbf{f}_2], \tag{5}$$

where λ_1, $\lambda_2 \in \mathbb{R}$. The operator $\mathscr{F}_{\mathbf{x}}$ can for example be a linear transform $\mathscr{F}_{\mathbf{x}}[\mathbf{f}] = \mathbf{C}\mathbf{f}(\mathbf{x})$ which together with the constraint (4) forces a certain linear combination of the outputs to be linearly dependent.

The operator $\mathscr{F}_{\mathbf{x}}$ could also include other linear operations on the function $\mathbf{f}(\mathbf{x})$. For example, we might know that the function $\mathbf{f}(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}^2$ should obey a certain partial differential equation $\mathscr{F}_{\mathbf{x}}[\mathbf{f}] = \frac{\partial f_1}{\partial x_1} + \frac{\partial f_2}{\partial x_2}$. A few more linear operators are listed in the Supplementary material, including integration as one of the most well-known.

The constraints (4) can either come from known physical laws or other prior knowledge of the process generating the data. Our objective is to encode these constraints in the mean and covariance functions (3) such that any sample from the corresponding GP prior (2) always obeys the constraint (4).

3 Building a constrained Gaussian process

3.1 Approach based on artificial observations

Just as Gaussian distributions are closed under linear transformations, so are GPs closed under linear operations (see Section 9.2). This can be used for a straightforward way of embedding linear operator constraints of the form (4) into GP regression. The idea is to treat the constraints as noise-free artificial observations $\{\tilde{\mathbf{x}}_k, \tilde{\mathbf{y}}_k\}_{k=1}^{\tilde{N}}$ with $\tilde{\mathbf{y}}_k = \mathbf{0}$ for all $k = 1 \ldots \tilde{N}$. The regression is then performed on the model $\tilde{\mathbf{y}}_k = \boldsymbol{\mathscr{F}}_{\tilde{\mathbf{x}}_k}[\mathbf{f}]$, where $\tilde{\mathbf{x}}_k$ are input points in the domain of interest. For example, one could let these artificial inputs $\tilde{\mathbf{x}}_k$ coincide with the points of prediction.

An advantage of this approach is that it allows constraints of the type (4) with a non-zero right hand side. Furthermore, there is no theoretical limit on how many constraints we can include (i.e. number of rows in $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$) – although in practice, of course, there is.

However, this is problematic mainly for two reasons. First of all, it makes the problem size grow. This increases memory requirements and execution time, and the numerical stability is worsen due to an increased condition number. This is especially clear from the fact that we want these observations to be noise-free, since the noise usually has a regularising effect. Secondly, the constraints are only enforced point-wise, so a sample drawn from the posterior fulfils the constraint only in our chosen points. The obvious way of compensating for this is by increasing the number of points in which the constraints are observed – but that exacerbates the first problem. Clearly, the challenge grows quickly with the dimension of the inferred function.

Embedding the constraints in the covariance function removes these issues – it makes the enforcement continuous while the problem size is left unchanged. We will now address the question of how to design such a covariance function.

3.2 A new construction

We want to find a GP prior (2) such that any sample $\mathbf{f}(\mathbf{x})$ from that prior obeys the constraints (4). In turn, this leads to constraints on the mean and covariance functions (3) of that prior. However, instead of posing these constraints on the mean and covariance functions directly, we consider $\mathbf{f}(\mathbf{x})$ to be related to another function $\mathbf{g}(\mathbf{x})$ via some operator $\boldsymbol{\mathscr{G}}_{\mathbf{x}}$

$$\mathbf{f}(\mathbf{x}) = \boldsymbol{\mathscr{G}}_{\mathbf{x}}[\mathbf{g}]. \tag{6}$$

The constraints (4) then amounts to

$$\boldsymbol{\mathscr{F}}_{\mathbf{x}}[\boldsymbol{\mathscr{G}}_{\mathbf{x}}[\mathbf{g}]] = 0. \tag{7}$$

We would like this relation to be true for any function $\mathbf{g}(\mathbf{x})$. To do that, we will interpret $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ and $\boldsymbol{\mathscr{G}}_{\mathbf{x}}$ as matrices and use a similar procedure to that of solving systems of linear equations. Since $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ and $\boldsymbol{\mathscr{G}}_{\mathbf{x}}$ are linear operators, we can think of $\boldsymbol{\mathscr{F}}_{\mathbf{x}}[\mathbf{f}]$ and $\boldsymbol{\mathscr{G}}_{\mathbf{x}}[\mathbf{g}]$ as matrix-vector multiplications where $\boldsymbol{\mathscr{F}}_{\mathbf{x}}[\mathbf{f}] = \boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{f}$, with $(\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{f})_i = \sum_{j=1}^{K} (\mathscr{F}_{\mathbf{x}})_{ij} f_j$ where each element $(\mathscr{F}_{\mathbf{x}})_{ij}$

in the operator matrix $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ is a scalar operator. With this notation, (7) can be written as

$$\boldsymbol{\mathscr{F}}_{\mathbf{x}}\boldsymbol{\mathscr{G}}_{\mathbf{x}} = 0. \tag{8}$$

This reformulation imposes constraints on the operator $\boldsymbol{\mathscr{G}}_{\mathbf{x}}$ rather than on the GP prior for $\mathbf{f}(\mathbf{x})$ directly. We can now proceed by designing a GP prior for $\mathbf{g}(\mathbf{x})$ and transform it using the mapping (6). We further know that GPs are closed under linear operations. More specifically, if $\mathbf{g}(\mathbf{x})$ is modelled as a GP with mean $\boldsymbol{\mu}_{\mathbf{g}}(\mathbf{x})$ and covariance $\mathbf{K}_{\mathbf{g}}(\mathbf{x}, \mathbf{x}')$, then $\mathbf{f}(\mathbf{x})$ is also a GP with

$$\mathbf{f}(\mathbf{x}) = \boldsymbol{\mathscr{G}}_{\mathbf{x}}\mathbf{g} \sim \mathcal{GP}\left(\boldsymbol{\mathscr{G}}_{\mathbf{x}}\, \boldsymbol{\mu}_{\mathbf{g}},\ \boldsymbol{\mathscr{G}}_{\mathbf{x}}\mathbf{K}_{\mathbf{g}}\boldsymbol{\mathscr{G}}_{\mathbf{x}'}^{\mathsf{T}}\right). \tag{9}$$

We use $(\boldsymbol{\mathscr{G}}_{\mathbf{x}}\mathbf{K}_{\mathbf{g}}\boldsymbol{\mathscr{G}}_{\mathbf{x}'}^{\mathsf{T}})_{ij}$ to denote that $(\boldsymbol{\mathscr{G}}_{\mathbf{x}}\mathbf{K}_{\mathbf{g}}\boldsymbol{\mathscr{G}}_{\mathbf{x}'}^{\mathsf{T}})_{ij} = (\mathscr{G}_{\mathbf{x}})_{ik}(\mathscr{G}_{\mathbf{x}'})_{jl}(K_{\mathbf{g}})_{kl}$, where $\boldsymbol{\mathscr{G}}_{\mathbf{x}}$ and $\boldsymbol{\mathscr{G}}_{\mathbf{x}'}$ act on the first and second argument of $\mathbf{K}_{\mathbf{g}}(\mathbf{x}, \mathbf{x}')$, respectively. See Section 9.2 for further details on linear operations on GPs.

The procedure to find the desired GP prior for $\mathbf{f}$ can now be divided into the following three steps

1. Find an operator $\mathscr{G}_{\mathbf{x}}$ that fulfils the condition (7).
2. Choose a mean and covariance function for $\mathbf{g}(\mathbf{x})$.
3. Find the mean and covariance functions for $\mathbf{f}(\mathbf{x})$ according to (9).

In addition to being resistant to the disadvantages of the approach described in Section 3.1, there are some additional strengths worth pointing out with this method. First of all, we have separated the task of encoding the constraints and encoding other desired properties of the kernel. The constraints are encoded in $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ and the remaining properties are determined by the prior for $\mathbf{g}(\mathbf{x})$, such as smoothness assumptions. Hence, satisfying the constraints does not sacrifice any desired behaviour of the target function.

Secondly, $\mathbf{K}(\mathbf{x}, \mathbf{x}')$ is guaranteed to be a valid covariance function provided that $\mathbf{K}_{\mathbf{g}}(\mathbf{x}, \mathbf{x}')$ is, since GPs are closed under linear functional transformations. From (9), it is clear that each column of $\mathbf{K}$ must fulfil all constraints encoded in $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$. Possibly $\mathbf{K}$ could be constructed only with this knowledge, assuming a general form and solving the resulting equation system. However, a solution may not just be hard to find, but one must also make sure that it is indeed a valid covariance function.

Furthermore, this approach provides a simple and straightforward way of constructing the covariance function even if the constraints have a complicated form. It makes no difference if the linear operators relate the components of the target function explicitly or implicitly – the procedure remains the same.

3.3 Illustrating example

We will now illustrate the method using the example (1) introduced already in the introduction. Consider a function $\mathbf{f}(\mathbf{x}) : \mathbb{R}^2 \mapsto \mathbb{R}^2$ satisfying $\frac{\partial f_1}{\partial x_1} + \frac{\partial f_2}{\partial x_2} = 0$, where $\mathbf{x} = [x_1,\ x_2]^{\mathsf{T}}$ and $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}),\ f_2(\mathbf{x})]^{\mathsf{T}}$. This equation describes all two-dimensional divergence-free vector fields. The constraint can be written as a linear constraint on the form (4) where $\boldsymbol{\mathcal{F}}_{\mathbf{x}} = [\frac{\partial}{\partial x_1}\ \frac{\partial}{\partial x_2}]$ and $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x})\ f_2(\mathbf{x})]^{\mathsf{T}}$. Modelling this function with a GP and building the covariance structure as described above, we first need to find the transformation $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$ such that (8) is fulfilled. For example, we could pick

$$\boldsymbol{\mathcal{G}}_{\mathbf{x}} = \begin{bmatrix} -\frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_1} \end{bmatrix}^{\mathsf{T}}. \tag{10}$$

If the underlying function $g(\mathbf{x}) : \mathbb{R}^2 \mapsto \mathbb{R}$ is given by $g(\mathbf{x}) \sim \mathcal{GP}\big(0, k_g(\mathbf{x}, \mathbf{x}')\big)$, then we can make use of (9) to obtain $\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}\big(\mathbf{0}, \mathbf{K}(\mathbf{x}, \mathbf{x}')\big)$ where

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \boldsymbol{\mathcal{G}}_{\mathbf{x}} k_g(\mathbf{x}, \mathbf{x}') \boldsymbol{\mathcal{G}}_{\mathbf{x}}^{\mathsf{T}} = \begin{bmatrix} \frac{\partial^2}{\partial x_2 x_2'} & -\frac{\partial^2}{\partial x_2 x_1'} \\ -\frac{\partial^2}{\partial x_1 x_2'} & \frac{\partial^2}{\partial x_1 x_1'} \end{bmatrix} k_g(\mathbf{x}, \mathbf{x}').$$

Using a covariance function with the following structure, we know that the constraint will be fulfilled by any function generated from the corresponding GP.

4 Finding the operator $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$

In a general setting it might be hard to find an operator $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$ that fulfils the constraint (8). Ultimately, we want an algorithm that can construct $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$ from a given $\boldsymbol{\mathcal{F}}_{\mathbf{x}}$. In more formal terms, the function $\boldsymbol{\mathcal{G}}_{\mathbf{x}}\mathbf{g}$ forms the nullspace of $\boldsymbol{\mathcal{F}}_{\mathbf{x}}$. The concept of nullspaces for linear operators is well-established [12], and does in many ways relate to real-number linear algebra.

However, an important difference is illustrated by considering a one-dimensional function $f(x)$ subject to the constraint $\mathcal{F}_x f = 0$ where $\mathcal{F}_x = \frac{\partial}{\partial x}$. The solution to this differential equation can not be expressed in terms of an arbitrary underlying function, but it requires $f(x)$ to be constant. Hence, the nullspace of $\frac{\partial}{\partial x}$ consists of the set of horizontal lines. Compare this with the real number equation $ab = 0$, $a \neq 0$, which is true only if $b = 0$. Since the nullspace differs between operators, we must be careful when discussing the properties of $\boldsymbol{\mathcal{F}}_{\mathbf{x}}$ and $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$ based on knowledge from real-number algebra.

Let us denote the rows in $\boldsymbol{\mathcal{F}}_{\mathbf{x}}$ as $\boldsymbol{f}_1^{\mathsf{T}}, \ldots, \boldsymbol{f}_L^{\mathsf{T}}$. We now want to find all solutions $\boldsymbol{g}$ such that

$$\boldsymbol{\mathcal{F}}_{\mathbf{x}} \boldsymbol{g} = \mathbf{0} \quad \Rightarrow \quad \boldsymbol{f}_i^{\mathsf{T}} \boldsymbol{g} = 0, \quad \forall \quad i = 1, \ldots, L. \tag{11}$$

```
Algorithm 1 Constructing $\mathscr{G}_{\mathbf{x}}$
Input: Operator matrix $\mathscr{F}_{\mathbf{x}}$
Output: Operator matrix $\mathscr{G}_{\mathbf{x}}$ where $\mathscr{F}_{\mathbf{x}}\mathscr{G}_{\mathbf{x}} = \mathbf{0}$
  Step 1: Make an ansatz $\boldsymbol{g} = \Gamma\boldsymbol{\xi}^{\boldsymbol{g}}$ for the columns in $\mathscr{G}_{\mathbf{x}}$.
  Step 2: Expand $\mathscr{F}_{\mathbf{x}}\Gamma\boldsymbol{\xi}^{\boldsymbol{g}}$ and collect terms.
  Step 3: Construct $\mathbf{A} \cdot \mathrm{vec}(\Gamma) = \mathbf{0}$ and find the vectors $\Gamma_1 \ldots \Gamma_P$ spanning its nullspace.
  Step 4: If $P = 0$, go back to Step 1 and make a new ansatz, i.e. extend the set of operators.
  Step 5: Construct $\mathscr{G}_{\mathbf{x}} = [\Gamma_1\boldsymbol{\xi}^{\boldsymbol{g}}, \ldots, \Gamma_P\boldsymbol{\xi}^{\boldsymbol{g}}]$.
```

The solutions $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_P$ to (11) will then be the columns of $\mathscr{G}_{\mathbf{x}}$. Each row vector $\boldsymbol{f}_j$ can be written as $\boldsymbol{f}_i = \Phi_i\boldsymbol{\xi}^{\boldsymbol{f}}$ where $\Phi_i \in \mathbb{R}^{K \times M_{\boldsymbol{f}}}$ and $\boldsymbol{\xi}^{\boldsymbol{f}} = [\xi_1, \ldots, \xi_{M_{\boldsymbol{f}}}]^{\mathsf{T}}$ is a vector of $M_{\boldsymbol{f}}$ scalar operators included in $\mathscr{F}_{\mathbf{x}}$. We now assume that $\boldsymbol{g}$ also can be written in a similar form $\boldsymbol{g} = \Gamma\boldsymbol{\xi}^{\boldsymbol{g}}$ where $\Gamma \in \mathbb{R}^{K \times M_{\boldsymbol{g}}}$ and $\boldsymbol{\xi}^{\boldsymbol{g}} = [\xi_1, \ldots, \xi_{M_{\boldsymbol{g}}}]^{\mathsf{T}}$ is a vector of $M_{\boldsymbol{g}}$ scalar operators. One may make the assumption that the same set of operators that are used to describe $\boldsymbol{f}_i$ also can be used to describe $\boldsymbol{g}$, i.e., $\boldsymbol{\xi}^{\boldsymbol{g}} = \boldsymbol{\xi}^{\boldsymbol{f}}$. However, this assumption might need to be relaxed. The constraints (11) can then be written as

$$(\boldsymbol{\xi}^{\boldsymbol{f}})^{\mathsf{T}}\Phi_i\Gamma\boldsymbol{\xi}^{\boldsymbol{g}} = 0, \qquad \forall \quad i = 1, \ldots, L. \tag{12}$$

We perform the multiplication and collect the terms in $\boldsymbol{\xi}^{\boldsymbol{f}}$ and $\boldsymbol{\xi}^{\boldsymbol{g}}$. The condition (12) then results in conditions on the parameters in Γ resulting a in a homogeneous system of linear equations

$$\mathbf{A} \cdot \mathrm{vec}(\Gamma) = \mathbf{0}. \tag{13}$$

The vectors $\mathrm{vec}(\Gamma_1), \ldots, \mathrm{vec}(\Gamma_P)$ spanning the nullspace of $\mathbf{A}$ in (13) are then used to compute the columns in $\mathscr{G}_{\mathbf{x}} = [\boldsymbol{g}_1, \ldots \boldsymbol{g}_P]$ where $\boldsymbol{g}_p = \Gamma_p\boldsymbol{\xi}^{\boldsymbol{g}}$. If it turns out that the nullspace of $\mathbf{A}$ is empty, one should start over with a new ansatz and extend the set of operators in $\boldsymbol{\xi}^{\boldsymbol{g}}$.

The outline of the procedure as described above is summarised in Algorithm 1. The algorithm is based upon a parametric ansatz rather than directly upon the theory for linear operators. Not only is it more intuitive, but it does also remove any conceptual challenges that theory may provide. A problem with this is that one may have to iterate before having found the appropriate set of operators in $\mathscr{G}_{\mathbf{x}}$. It might be of interest to examine possible alternatives to this algorithm that does not use a parametric approach. Let us now illustrate the method with an example.

4.1 Divergence-free example revisited

Let us return to the example discussed in Section 3.3, and show how the solution found by visual inspection also can be found with the algorithm described above. Since $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ only contains first-order derivative operators, we assume that a column in $\boldsymbol{\mathscr{G}}_{\mathbf{x}}$ does so as well. Hence, let us propose the following ansatz (step 1)

$$\boldsymbol{g} = \begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \end{bmatrix} = \Gamma \boldsymbol{\xi}^{\boldsymbol{g}}. \tag{14}$$

Applying the constraint, expanding and collecting terms (step 2) we find

$$\begin{aligned} \boldsymbol{\mathscr{F}}_{\mathbf{x}} \Gamma \boldsymbol{\xi}^{\boldsymbol{g}} &= \begin{bmatrix} \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} \end{bmatrix} \begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \end{bmatrix} \\ &= \gamma_{11} \frac{\partial^2}{\partial x_1^2} + (\gamma_{12} + \gamma_{21}) \frac{\partial^2}{\partial x_1 \partial x_2} + \gamma_{22} \frac{\partial^2}{\partial x_2^2}, \end{aligned} \tag{15}$$

where we have used the fact that $\frac{\partial^2}{\partial x_i \partial x_j} = \frac{\partial^2}{\partial x_j \partial x_i}$ assuming continuous second derivatives. The expression (15) equals zero if

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \gamma_{11} \\ \gamma_{12} \\ \gamma_{21} \\ \gamma_{22} \end{bmatrix} = \mathbf{A} \cdot \operatorname{vec}(\Gamma) = \mathbf{0}. \tag{16}$$

The nullspace is spanned by a single vector (step 3)

$$\begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{21} & \gamma_{22} \end{bmatrix}^{\mathsf{T}} = \lambda \begin{bmatrix} 0 & -1 & 1 & 0 \end{bmatrix}^{\mathsf{T}}, \quad \lambda \in \mathbb{R}. \tag{17}$$

Choosing $\lambda = 1$, we get $\boldsymbol{\mathscr{G}}_{\mathbf{x}} = \begin{bmatrix} -\frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_1} \end{bmatrix}^{\mathsf{T}}$ (step 5), which is the same as in (10).

4.2 Generalisation

Although there are no conceptual problems with the algorithm introduced above, the procedure of expanding and collecting terms appears a bit informal. In a general form, the algorithm is reformulated such that the operators are completely left out from the solution process. The drawback of this is a more cumbersome notation, and we have therefore limited the presentation to this simplified version. However, the general algorithm is found in the Supplementary material of this paper.

5 Experimental results

5.1 Simulated divergence-free function

Consider the example in Section 3.3. An example of a function fulfilling $\frac{\partial f_1}{\partial x_1} + \frac{\partial f_2}{\partial x_2} = 0$ is

$$\begin{aligned} f_1(x_1, x_2) &= e^{-ax_1x_2}\big(ax_1 \sin(x_1x_2) - x_1 \cos(x_1x_2)\big), \\ f_2(x_1, x_2) &= e^{-ax_1x_2}\big(x_2 \cos(x_1x_2) - ax_2 \sin(x_1x_2)\big), \end{aligned} \tag{18}$$

where a denotes a constant. We will now study how the regression of this function differs when using the covariance function found in Section 3.3 as compared to a diagonal covariance function $\mathbf{K}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')I$. The measurements generated are corrupted with Gaussian noise such that $\mathbf{y}_k = \mathbf{f}(\mathbf{x}_k) + \mathbf{e}_k$, where $\mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$. The squared exponential covariance function $k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left[-\frac{1}{2}l^{-2}\|\mathbf{x} - \mathbf{x}'\|^2\right]$ has been used for k_g and k with hyperparameters chosen by maximising the marginal likelihood. We have used the value $a = 0.01$ in (18).

We have used 50 measurements randomly picked over the domain $[0\ \ 4] \times [0\ \ 4]$, generated with the noise level $\sigma = 10^{-4}$. The points for prediction corresponds to a discretisation using 20 uniformly distributed points in each direction, and hence a total of $N_P = 20^2 = 400$. We have included the approach described is Section 3.1 for comparison. The number of artificial observations have been chosen as random subsets of the prediction points, up to and including the full set.

The comparison is made with regard to the root mean squared error $e_{\text{rms}} = \sqrt{\frac{1}{N_P}\bar{\mathbf{f}}_\Delta^\mathsf{T}\bar{\mathbf{f}}_\Delta}$, where $\bar{\mathbf{f}}_\Delta = \hat{\bar{\mathbf{f}}} - \bar{\mathbf{f}}$ and $\bar{\mathbf{f}}$ is a concatenated vector storing the true function values in all prediction points and $\hat{\bar{\mathbf{f}}}$ denotes the reconstructed equivalent. To decrease the impact of randomness, each error value has been formed as an average over 50 reconstructions given different sets of measurements.

An example of the true field, measured values and reconstruction errors using the different methods is seen in Figure 2. The result from the experiment is seen in Figure 3a. Note that the error from the approach with artificial observations is decreasing as the number of observations is increased, but only to a certain point. Have in mind, however, that the Gram matrix is growing, making the problem larger and worse conditioned. The result from our approach is clearly better, while the problem size is kept small and numerical problems are therefore avoided.

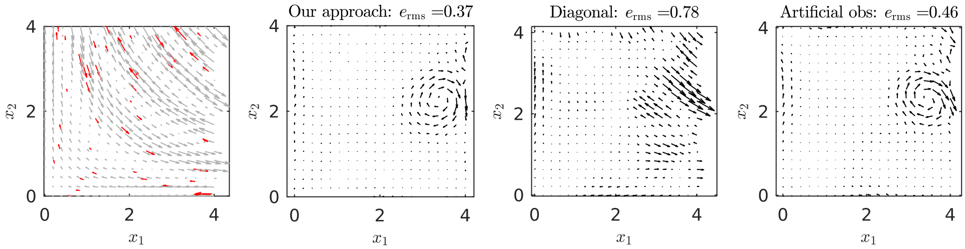


Figure 2: Left: Example of field plots illustrating the measurements (red arrows) and the true field (grey arrows). Remaining three plots: reconstructed fields subtracted from the true field. The artificial observations of the constraint have been made in the same points as the predictions are made.

5.2 Real data experiment

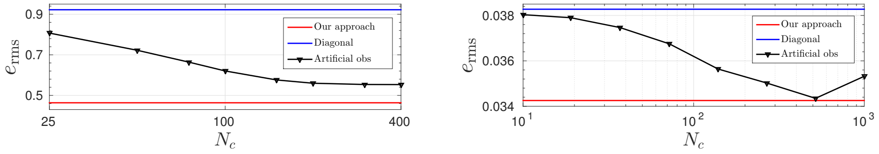
Magnetic fields can mathematically be considered as a vector field mapping a 3D position to a 3D magnetic field strength. Based on the magnetostatic equations, this can be modelled as a curl-free vector field. Following Section 9.3, our method can be used to encode the constraints in the following covariance function (which also has been presented elsewhere [24])

$$\mathbf{K}_{\text{curl}}(\mathbf{x},\mathbf{x}') = \sigma_f^2 e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l^2}} \left(\mathbf{I}_3 - \left(\frac{\mathbf{x}-\mathbf{x}'}{l}\right)\left(\frac{\mathbf{x}-\mathbf{x}'}{l}\right)^{\mathsf{T}} \right). \tag{19}$$

With a magnetic sensor and an optical positioning system, both position and magnetic field data have been collected in a magnetically distorted indoor environment, see the Supplementary material for details about the experimental details. In Figure 1 the predicted magnitude of the magnetic field over a two-dimensional domain for three different heights above the floor is displayed. The predictions have been made based on 500 measurements sampled from the trajectory given by the blue curve.

Similar to the simulated experiment in Section 5.1, we compare the predictions of the curl-free covariance function (19) with the diagonal covariance function and the diagonal covariance function using artificial observations. The results have been formed by averaging the error over 50 reconstructions. In each iteration, training data and test data were randomly selected from the data set collected in the experiment. 500 train data points and 1 000 test data points were used.

The result is seen in Figure 3b. We recognise the same behaviour as we saw for the simulated experiment in Figure 3a. Note that the accuracy of the artificial observation approach gets very close to our approach for a large number of artificial observations. However, in the last step of increasing the



(a) Simulated experiment

(b) Real-data experiment

Figure 3: Accuracy of the different approaches as the number of artificial observations N_c is increased.

artificial observations, the accuracy decreases. This is probably caused by the numerical errors that follows from an ill-conditioned Gram matrix.

6 Related work

Many problems in which GPs are used contain some kind of constraint that could be well exploited to improve the quality of the solution. Since there are a variety of ways in which constraints may appear and take form, there is also a variety of methods to deal with them. The treatment of inequality constraints in GP regression have been considered for instance in [1] and [5], based on local representations in a limited set of points. The paper [13] proposes a finite-dimensional GP-approximation to allow for inequality constraints in the entire domain.

It has been shown that linear constraints satisfied by the training data will be satisfied by the GP prediction as well [21]. The same paper shows how this result can be extended to quadratic forms through a parametric reformulation and minimisation of the Frobenious norm, with application demonstrated for pose estimation. Another approach on capturing human body features is described in [20], where a face-shape model is included in the GP framework to imply anatomic correctness. A rigorous theoretical analysis of degeneracy and invariance properties of Gaussian random fields is found in [8], including application examples for one-dimensional GP problems.

Although constraints in most situations are formulated on the outputs of the GP, there are also situations in which they are acting on the inputs. An example of this is given in [23], describing a method of benefit from ordering constraints on the input to reduce the negative impact of input noise.

Applications within medicine include gene-disease association through functional expectation constraints [11] and lung disease sub-type identification using a mixture of GPs and constraints encoded with Markov random fields [19]. Another way of viewing constraints is as modified prior distributions. By making use of the so-called multivariate generalised von Mises distribution, [14] ends up in a version of GP regression customised for circular variable problems. Other fields of interest include using GPs in approximately solving one-dimensional partial differential equations [9, 15, 16].

Generally speaking, the papers mentioned above consider problems in which the constraints are dealt with using some kind of external enforcement – that is, they are not explicitly incorporated into the model, but rely on approximations or finite representations. Therefore, the constraints may just be approximately satisfied and not necessarily in a continuous manner, which differs from the method proposed in this paper. Of course, comparisons can not be done directly between methods that have been developed for different kinds of constraints. The interest in this paper is multivariate problems where the constraints are linear combinations of the outputs that are known to equal zero.

For multivariate problems, constructing the covariance function is particularly challenging due to the correlation between the output components. We refer to [2] for a very useful review. The basic idea behind the so-called *separable kernels* is to separate the process of modelling the covariance function for each component and the process of modelling the correlation between them. The final covariance function is chosen for example according to some method of regularisation. Another class of covariance functions is the *invariant kernels*. Here, the correlation is inherited from a known mathematical relation. The curl- and divergence free covariance functions are such examples where the structure follows directly from the underlying physics, and has been shown to improve the accuracy notably for regression problems [24]. Another example is the method proposed in [4], where the Taylor expansion is used to construct a covariance model given a known relationship between the outputs. A very useful property on linear transformations is given in [22], based on the GPs natural inheritance of features imposed by linear operators. This fact has for example been used in developing a method for monitoring infectious diseases [3].

The method proposed in this work is exploiting the transformation property to build a covariance function of the invariant kind for a multivariate GP. We show how this property can be exploited to incorporate knowledge of linear constraints into the covariance function. Moreover, we present an algorithm of constructing the required transformation. This way, the constraints are built into the prior and are guaranteed to be fulfilled in the entire domain.

7 Conclusion and future work

We have presented a method for designing the covariance function of a multivariate Gaussian process subject to known linear operator constraints on the target function. The method will by construction guarantee that any sample drawn from the resulting process will obey the constraints in all points. Numerical simulations show the benefits of this method as compared to alternative approaches. Furthermore, it has been demonstrated to improve the performance on real data as well.

As mentioned in Section 4, it would be desirable to describe the requirements on $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$ more rigorously. That might allow us to reformulate the construction algorithm for $\boldsymbol{\mathcal{G}}_{\mathbf{x}}$ in a way that allows for a more straightforward approach as compared to the parametric ansatz that we have proposed. In particular, our method relies upon the requirement that the target function can be expressed in terms of an underlying *potential* function $\mathbf{g}$. This leads to the intriguing and nontrivial question: Is it possible to mathematically guarantee the existence of such a potential? If the answer to this question is yes, the next question will of course be what it look like and how it relates to the target function.

Another possible topic of further research is the extension to constraints including *nonlinear* operators, which for example might rely upon a linearisation in the domain of interest. Furthermore, it may be of potential interest to study the extension to a non-zero right-hand side of (4).

8 Acknowledgements

This research is financially supported by the Swedish Foundation for Strategic Research (SSF) via the project *ASSEMBLE* (Contract number: RIT 15-0012). The work is also supported by the Swedish Research Council (VR) via the project *Probabilistic modelling of dynamical systems* (Contract number: 621-2013-5524). We are grateful for the help and equipment provided by the UAS Technologies Lab, Artificial Intelligence and Integrated Computer Systems Division (AIICS) at the Department of Computer and Information Science (IDA), Linköping University, Sweden. The real data set used in this paper has been collected by some of the authors together with Manon Kok, Arno Solin, and Simo Särkkä. We thank them for allowing us to use this data. We also thank Manon Kok for supporting us with the data processing. Furthermore, we would like to thank Carl Rasmussen and Marc Deisenroth for fruitful discussions on constrained GPs.

9 Supplementary material

9.1 Linear operators

In this work we consider linear operators on functions. Such an operator transforms a function $\mathbf{f}(\mathbf{x})$ to another function $\mathbf{g}(\mathbf{z})$. We denote this according to

$$\mathbf{g}(\mathbf{z}) = \boldsymbol{\mathscr{F}}_{\mathbf{z}}[\mathbf{f}(\mathbf{x})]. \tag{20}$$

This linear operator could be *differentiation* of a function. If $D = 1$ and $K = 1$ this will be defined as

$$g(z) = \mathscr{F}_z[f] = \frac{\partial f(x)}{\partial x}\Big|_{x=z}, \tag{21a}$$

which slightly more informal also can be written as

$$g(x) = \mathscr{F}_x[f] = \frac{\partial f(x)}{\partial x}. \tag{21b}$$

Also *integration* of a scalar function $f(x)$ over an interval $[z_1, z_2]$ is a linear operator

$$g(\mathbf{z}) = \mathscr{F}_{\mathbf{z}}[f] = \int_{z_1}^{z_2} f(x)dx, \tag{22}$$

where $g(\mathbf{z})$ is a scalar-valued function with a two-dimensional input $\mathbf{z} = [z_1,\ z_2]^{\mathsf{T}}$. Note that in the two examples given above, the inputs of f and g will not be the same, not even of the same dimension!

Input wrapping is another way to construct new covariance functions from old ones [18, page 92]. It utilises a nonlinear wrapping $\mathbf{x} = \mathbf{u}(\mathbf{z})$ of the input variables. This wrapping can also be considered as a linear operator, where

$$\mathbf{g}(\mathbf{z}) = \boldsymbol{\mathscr{F}}_{\mathbf{z}}[\mathbf{f}] = \mathbf{f}(\mathbf{x})|_{\mathbf{x}=\mathbf{u}(\mathbf{z})}. \tag{23}$$

This operator also changes the function input and possibly also its dimension. Even though the wrapping itself might be nonlinear, the operator corresponding to this wrapping is in fact linear.

It is straightforward to show that all three operators presented above do fulfil the linearity condition.

9.2 Gaussian processes under linear operations

It is well-known that Gaussian distributions are closed under linear transformation. In similar manner, Gaussian processes are closed under linear operations [6, 10, 17, 18].

By applying the functional $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ on both the mean function and the covariance function, the GP prior for $\boldsymbol{\mathscr{F}}_{\mathbf{x}}$ is given by

$$\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{f} \sim \mathcal{GP}\left(\boldsymbol{\mathscr{F}}_{\mathbf{x}}\ \boldsymbol{\mu},\ \mathrm{Cov}\left[\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{f}(\mathbf{x}),\ \boldsymbol{\mathscr{F}}_{\mathbf{x}'}\mathbf{f}(\mathbf{x}')\right]\right). \tag{24}$$

The covariance becomes

$$\begin{aligned}
&\mathrm{Cov}\left[\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{f}(\mathbf{x}),\ \boldsymbol{\mathscr{F}}_{\mathbf{x}'}\mathbf{f}(\mathbf{x}')\right] \\
&= \mathbb{E}\left[\left(\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{f}(\mathbf{x}) - \boldsymbol{\mathscr{F}}_{\mathbf{x}}\boldsymbol{\mu}(\mathbf{x})\right)\left(\boldsymbol{\mathscr{F}}_{\mathbf{x}'}\mathbf{f}(\mathbf{x}') - \boldsymbol{\mathscr{F}}_{\mathbf{x}'}\boldsymbol{\mu}(\mathbf{x}')\right)^{\mathsf{T}}\right] \\
&= \boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbb{E}\left[\left(\mathbf{f}(\mathbf{x}) - \boldsymbol{\mu}(\mathbf{x})\right)\left(\mathbf{f}(\mathbf{x}') - \boldsymbol{\mu}(\mathbf{x}')\right)^{\mathsf{T}}\right]\boldsymbol{\mathscr{F}}_{\mathbf{x}'}^{\mathsf{T}} \\
&= \boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{K}\boldsymbol{\mathscr{F}}_{\mathbf{x}'}^{\mathsf{T}},
\end{aligned} \tag{25}$$

where by the notation $(\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{K}\boldsymbol{\mathscr{F}}_{\mathbf{x}'}^{\mathsf{T}})_{ij}$ we mean that

$$(\boldsymbol{\mathscr{F}}_{\mathbf{x}}\mathbf{K}\boldsymbol{\mathscr{F}}_{\mathbf{x}'}^{\mathsf{T}})_{ij} = (\mathscr{F}_{\mathbf{x}})_{ik}(\mathscr{F}_{\mathbf{x}'})_{jl}K_{kl}, \tag{26}$$

and where $(\mathscr{F}_{\mathbf{x}})_{ik}$ and $(\mathscr{F}_{\mathbf{x}'})_{jl}$ act on the first and second argument of $\mathbf{K}_{kl}(\mathbf{x}, \mathbf{x}')$, respectively.

We should point out that some care must be taken when applying this procedure. For example, if we would like to consider the derivative of a function governed by a GP, we must make sure that this function is modelled in a way such that the derivative actually exists. This may sound obvious, yet important to remember since the set of standard covariance functions includes members that are not differentiable – among those we find Matérn$_{1/2}$ [18].

9.3 Generalisation of Section 4

In this supplementary material we will generalise the method described on how to solve operator matrix equations on the form

$$\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{G}} = \mathbf{0},$$

where we want to find $\boldsymbol{\mathscr{G}}$ given $\boldsymbol{\mathscr{F}}$ [1]. If $\boldsymbol{\mathscr{F}} \in \mathbb{R}^{m\times n}$ is a real valued matrix, $\boldsymbol{\mathscr{G}}$ can easily be found by letting the columns in $\boldsymbol{\mathscr{G}}$ span the nullspace of $\boldsymbol{\mathscr{F}}$ (provided such a nullspace exist). However, if the elements of $\boldsymbol{\mathscr{F}}$ are operators, the situation is more tricky. This supplementary material generalises the parametric approach presented in Section 4 for arbitrary operators of any order. The strategy is to study the vector space of homogeneous polynomials where the operators are interpreted as the variables of these polynomials.

In Section 9.3, we assume that both $\boldsymbol{\mathscr{F}}$ and $\boldsymbol{\mathscr{G}}$ consist of first order operators and in Section 9.3 we generalise this to allow for any order of the operators.

[1] In this supplementary material, the argument $\mathbf{x}$ is omitted for simplified notation

First order operator equation

Consider the matrix $\boldsymbol{\mathscr{F}} \in \mathcal{P}_p^{m\times n}$, where $\mathcal{P}_p$ is a vector space of first order operators

$$\mathcal{P}_p = \{a_1 y_1 + \ldots a_p y_p | a_1, \ldots, a_p \in \mathbb{R}\}, \tag{27}$$

where $y_1, \ldots, y_p$ is the basis in that vector space. The basis components y_k can for example represent derivative operators $y_k = \frac{\partial}{\partial x_k}$. We want to find the vectors $\boldsymbol{\mathscr{g}} \in \mathcal{P}_p^n$ such that $\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{g}} = \mathbf{0}$ is fulfilled. We can write $\boldsymbol{\mathscr{F}} \in \mathcal{P}_p^{m\times n}$ and $\boldsymbol{\mathscr{g}} \in \mathcal{P}_p^n$ as

$$\mathscr{F}_{ij} = \sum_{k=1}^{p} \phi_{ijk} y_k, \qquad \phi_{ijk} = \{\Phi\}_{ijk} \in \mathbb{R}, \tag{28a}$$

$$\mathscr{g}_j = \sum_{k=1}^{p} \gamma_{jk} y_k, \qquad \gamma_{jk} = \{\Gamma\}_{jk} \in \mathbb{R}, \tag{28b}$$

where $\Phi \in \mathbb{R}^{m\times n\times p}$ and $\Gamma \in \mathbb{R}^{n\times p}$. This gives

$$\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{g}} = 0 \Leftrightarrow \sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{p} \phi_{ijk} y_k \gamma_{jl} y_l = 0 \quad \forall\ i = 1 : m. \tag{29}$$

For each i, we have a quadratic form

$$\mathbf{y}^{\mathsf{T}} \Phi_i \Gamma \mathbf{y} = 0, \tag{30}$$

where $\Phi_i \in \mathbb{R}^{p\times n}$ with $\{\Phi_i\}_{kj} = \phi_{ijk}$ and $\Gamma \in \mathbb{R}^{n\times p}$ with $\{\Gamma\}_{jk} = \gamma_{jk}$.

The quadratic form is equal to zero for all $\mathbf{y}$ if and only if

$$\Phi_i \Gamma + \Gamma^{\mathsf{T}} \Phi_i^{\mathsf{T}} = 0 \quad \forall \quad i = 1 : m. \tag{31}$$

Example 1 (divergence free vector field)

We consider the following vector of operators $\boldsymbol{\mathscr{F}} \in \mathcal{P}_3^{1\times 3}$

$$\boldsymbol{\mathscr{F}} = \nabla_{\mathbf{x}} = \left[\frac{\partial}{\partial x_1}, \ \frac{\partial}{\partial x_2}, \ \frac{\partial}{\partial x_3}\right], \tag{32}$$

where

$$\mathscr{F}_{ij} = \sum_{k=1}^{3} \phi_{ijk} y_k, \quad \forall \quad i = 1, \quad j = 1, 2, 3, \tag{33}$$

where $y_k = \frac{\partial}{\partial x_k}$. Following the notation introduced above, for this particular operator matrix we have

$$\Phi_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{34}$$

We now want of find a vector $\boldsymbol{g} \in \mathcal{P}^3$ that fulfils $\boldsymbol{\mathcal{F}}\boldsymbol{g} = \mathbf{0}$ for all $\mathbf{y}$. We assume that this operator vector is in $\boldsymbol{g} \in \mathcal{P}_3^3$ and can be written

$$g_j = \sum_{k=1}^{3} \gamma_{jk} y_k \quad j = 1, 2, 3, \tag{35}$$

where $\Gamma \in \mathbb{R}^{3\times 3}$ is unknown. Now we have that

$$\Phi_1 \Gamma + \Gamma^\mathsf{T} \Phi_1^\mathsf{T} = 0 \tag{36a}$$

$$\Rightarrow \begin{bmatrix} \gamma_{11} & \gamma_{12} - \gamma_{21} & \gamma_{13} - \gamma_{31} \\ \gamma_{21} - \gamma_{12} & \gamma_{22} & \gamma_{23} - \gamma_{32} \\ \gamma_{31} - \gamma_{13} & \gamma_{32} - \gamma_{23} & \gamma_{33} \end{bmatrix} = 0, \tag{36b}$$

which in turn gives

$$\gamma_{11} = 0, \qquad \gamma_{12} + \gamma_{21} = 0, \tag{37a}$$

$$\gamma_{22} = 0, \qquad \gamma_{13} + \gamma_{31} = 0, \tag{37b}$$

$$\gamma_{33} = 0, \qquad \gamma_{23} + \gamma_{32} = 0. \tag{37c}$$

The nullspace of (36a) is then spanned by

$$\Gamma = \lambda_1 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} + \lambda_3 \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

which gives

$$\boldsymbol{g} = \lambda_1 \begin{bmatrix} 0 \\ \frac{\partial}{\partial x_3} \\ -\frac{\partial}{\partial x_2} \end{bmatrix} + \lambda_2 \begin{bmatrix} -\frac{\partial}{\partial x_3} \\ 0 \\ \frac{\partial}{\partial x_1} \end{bmatrix} + \lambda_3 \begin{bmatrix} \frac{\partial}{\partial x_2} \\ -\frac{\partial}{\partial x_1} \\ 0 \end{bmatrix}, \lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}.$$

Example 2 (curl free vector field)

We consider the following vector of operators $\boldsymbol{\mathcal{F}} \in \mathcal{P}_3^{3\times 3}$

$$\boldsymbol{\mathcal{F}} = \begin{bmatrix} 0 & \frac{\partial}{\partial x_3} & -\frac{\partial}{\partial x_2} \\ -\frac{\partial}{\partial x_3} & 0 & \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} & -\frac{\partial}{\partial x_1} & 0 \end{bmatrix}, \tag{38}$$

where

$$\mathscr{F}_{ij} = \sum_{k=1}^{3} \phi_{ijk} y_k, \quad \forall\ i = 1:3, \quad j = 1:3, \tag{39}$$

where $y_k = \frac{\partial}{\partial x_k}$. For this particular operator matrix we have

$$\Phi_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \ \Phi_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \ \Phi_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

We now want to find a vector $\boldsymbol{\mathscr{g}} \in \mathcal{P}^3$ which fulfils $\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{g}} = \mathbf{0}$ for all $\mathbf{y}$. We assume that this operator vector is in $\boldsymbol{\mathscr{g}} \in \mathcal{P}_3^3$ and can be written

$$\mathscr{g}_j = \sum_{k=1}^{3} \gamma_{jk} y_k \quad j = 1, 2, 3, \tag{40}$$

where $\Gamma \in \mathbb{R}^{3\times 3}$ is unknown. Now we have that

$$\Phi_1\Gamma + \Gamma^\mathsf{T}\Phi_1^\mathsf{T} = 0 \Rightarrow \begin{bmatrix} 0 & -\gamma_{31} & \gamma_{21} \\ -\gamma_{31} & -2\gamma_{32} & \gamma_{22}-\gamma_{33} \\ \gamma_{21} & \gamma_{22}-\gamma_{33} & 2\gamma_{23} \end{bmatrix} = 0,$$

$$\Phi_2\Gamma + \Gamma^\mathsf{T}\Phi_2^\mathsf{T} = 0 \Rightarrow \begin{bmatrix} 2\gamma_{31} & \gamma_{32} & \gamma_{33}-\gamma_{11} \\ \gamma_{32} & 0 & -\gamma_{12} \\ \gamma_{33}-\gamma_{11} & -\gamma_{12} & -2\gamma_{13} \end{bmatrix} = 0,$$

$$\Phi_3\Gamma + \Gamma^\mathsf{T}\Phi_3^\mathsf{T} = 0 \Rightarrow \begin{bmatrix} 2\gamma_{21} & \gamma_{22}-\gamma_{11} & \gamma_{23} \\ \gamma_{22}-\gamma_{11} & -2\gamma_{12} & -\gamma_{13} \\ \gamma_{23} & -\gamma_{13} & 0 \end{bmatrix} = 0,$$

which in turn gives

$$\gamma_{22} - \gamma_{33} = 0, \qquad \gamma_{23} = 0, \qquad \gamma_{32} = 0, \tag{41a}$$

$$\gamma_{33} - \gamma_{11} = 0, \qquad \gamma_{13} = 0, \qquad \gamma_{31} = 0, \tag{41b}$$

$$\gamma_{22} - \gamma_{11} = 0, \qquad \gamma_{12} = 0, \qquad \gamma_{21} = 0. \tag{41c}$$

The nullspace of (41a) is then spanned by the single base vector

$$\Gamma = \lambda_1 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \lambda_1 \in \mathbb{R}, \tag{42}$$

which gives

$$\boldsymbol{\mathscr{g}} = \lambda_1 \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} \end{bmatrix}, \quad \lambda_1 \in \mathbb{R}. \tag{43}$$

Figure 4: Three snapshots from the measurement collection. The senor platform was moved around by hand during approximately three minutes.

The final covariance function becomes

$$\mathbf{K}(\mathbf{x},\mathbf{x}') = \begin{bmatrix} \frac{\partial^2}{\partial x_1 \partial x_1'} & \frac{\partial^2}{\partial x_1 \partial x_2'} & \frac{\partial^2}{\partial x_1 \partial x_3'} \\ \frac{\partial^2}{\partial x_2 \partial x_1'} & \frac{\partial^2}{\partial x_2 \partial x_2'} & \frac{\partial^2}{\partial x_2 \partial x_3'} \\ \frac{\partial^2}{\partial x_3 \partial x_1'} & \frac{\partial^2}{\partial x_3 \partial x_2'} & \frac{\partial^2}{\partial x_3 \partial x_3'} \end{bmatrix} k_g(\mathbf{x},\mathbf{x}'). \tag{44}$$

If we use the squared exponential covariance function

$$k_g(\mathbf{x},\mathbf{x}') = \sigma_f^2 e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l^2}}, \tag{45}$$

we get

$$\mathbf{K}(\mathbf{x},\mathbf{x}') = \frac{\sigma_f^2}{l^2} e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l^2}} \left(\mathbf{I}_3 - \left(\frac{\mathbf{x}-\mathbf{x}'}{l}\right)\left(\frac{\mathbf{x}-\mathbf{x}'}{l}\right)^{\mathsf{T}} \right). \tag{46}$$

This covariance function is used in the real data experiment in Section 5.2. Note, that the version in the paper does not use l^2 in the denominator (which we also would get here if we would multiply (43) with l^2, still providing the same constraints).

Higher order operator equation

Now, consider the matrix $\boldsymbol{\mathscr{F}} \in \mathcal{P}_{p,q}^{m \times n}$, where $\mathcal{P}_{p,q}$ is a vector space of all homogeneous polynomials of degree q in p variables

$$\mathcal{P}_{p,q} = \left\{ \sum_{k_1}^{p} \cdots \sum_{k_q}^{p} a_{k_1,\ldots,k_q} y_{k_1} \cdots y_{k_q} | a_{k_1,\ldots,k_q} \in \mathbb{R} \right\},$$

where the nominals $y_{k_1} \cdots y_{k_q}$ constitute the basis of that vector space. The components y_k can for example represent derivative operators $y_k = \frac{\partial}{\partial x_k}$ and $\mathcal{P}_{p,q}$ then contain all qth order derivatives of $x_1 \ldots x_q$. We want to find the vectors $\boldsymbol{\mathscr{g}} \in \mathcal{P}^n_{p,q_g}$ such that $\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{g}} = \mathbf{0}$ is fulfilled. We can write $\boldsymbol{\mathscr{F}} \in \mathcal{P}^{m\times n}_{p,q}$ and $\boldsymbol{\mathscr{g}} \in \mathcal{P}^n_{p,q_g}$ as

$$\mathscr{F}_{ij} = \sum_{k_1}^{p} \cdots \sum_{k_q}^{p} \phi_{i,j,k_1,\dots,k_q} y_{k_1} \cdots y_{k_q}, \tag{47a}$$

$$\mathscr{g}_j = \sum_{k_1}^{p} \cdots \sum_{k_q}^{p} \gamma_{j,k_1,\dots,k_{q_g}} y_{k_1} \cdots y_{k_{q_g}}, \tag{47b}$$

where $\Phi \in \mathbb{R}^{m\times n\times p^{\times q}}$ and $\mathbf{b} \in \mathbb{R}^{n\times p^{\times q}}$ (here $p^{\times q}$ denotes $\underbrace{p \times \cdots \times p}_{q \text{ times}}$). This gives

$$\boldsymbol{\mathscr{F}}\boldsymbol{\mathscr{g}} = 0 \Leftrightarrow \sum_{j}^{n} \sum_{k_1}^{p} \cdots \sum_{k_q}^{p} \sum_{l_1}^{p} \cdots \sum_{l_q}^{p} \Big\{$$
$$\phi_{ijk_1\dots k_q} y_{k_1} \cdots y_{k_q} \gamma_{jl_1\dots l_{q_g}} y_{l_1} \cdots y_{l_{q_g}} \Big\} = 0 \quad \forall\ i = 1:m.$$

For each i, this is an algebraic form of order $q + q_g$

$$\sum_{j}^{n} \sum_{k_1\dots k_q, l_1\dots l_q \in \{d_1 \dots d_{q+q_g}\}} \phi_{ijd_1\dots d_q} \gamma_{jd_{q+1}\dots d_{q+q_g}} = 0 \quad \forall \quad i = 1:m,$$
$$k_1 = 1:p, \quad \dots, \quad k_q = 1:p,$$
$$l_1 = 1:p, \quad \dots, \quad l_q = 1:p,$$

where the second sum sums over all permutations of $k_1 \dots k_q, l_1 \dots l_q$.

9.4 Real data experiment description

This section contains more details about the real data experiment described in Section 5.2.

Experiment setup

To collect the measurements we made use of a wooden platform, see Figure 5. The platform was equipped with a Trivisio Colibri wireless IMU (TRIVISIO Prototyping GmbH, http://www.trivisio.com/), sampled at 100 Hz. The sensor includes both an accelerometer, a gyroscope, and a magnetometer. For additional validation a Google Nexus 5 smartphone was also mounted on the platform even tough its data was never used in this experiment.

On the platform, five markers were mounted. An optical reference system (Vicon) with several cameras mounted in the ceiling measured the 3D position of each marker, and hence also the position and the orientation of the platform relative to its predefined origin.

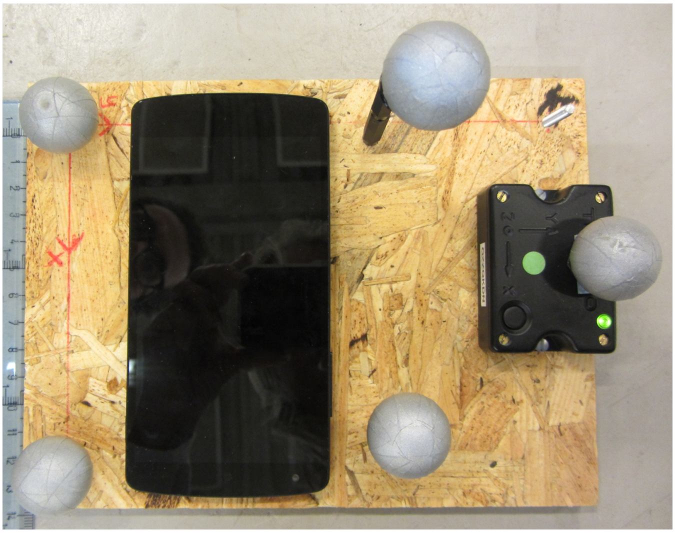


Figure 5: Platform with magnetic sensors. The sensor to the left is the Trivisio sensor, whose magnetometer data we used during the experiment. The platform was also equipped with multiple markers visible to an optical reference system (Vicon).

Experiment execution

The sensor platform was moved around by hand up and down in a volume of $4 \times 4 \times 2$ meters, see Figure 4. During the experiment, measurements were collected from the sensors on the platform as well from the optical reference system. The data from the different sensors were collected asynchronously. The experiment lasted for 187 seconds.

Pre-processing of data

The position and orientation data from the optical reference system was synchronised with the data from the Trivisio sensor. The synchronisation was performed based on correlation analysis of the angular velocities measured by both systems.

The position in global coordinates of the Trivisio sensor was computed based on the position data, the orientation data, and the displacement of the Trivisio sensor relative to the predefined origin of the platform.

The magnetometer data from the Trivisio sensor was rotated from sensor-fixed coordinates to global coordinates using the orientation data from the optical reference system. These rotated measurements describe the magnetic field in global coordinates at the sensor positions computed above. In Section 5.2 of the main paper, these position data and magnetic field data are considered as input and output data, respectively.

References

[1] P. Abrahamsen and F. E. Benth. “Kriging with inequality constraints.” In: *Math. Geol.* 33.6 (2001), pp. 719–744.

[2] M. A. Álvarez, L. Rosasco and N. D. Lawrence. “Kernels for Vector-Valued Functions: A Review”. In: *Foundations and Trends in Machine Learning* 4.3 (Mar. 2012), pp. 195–266.

[3] R. Andrade-Pacheco, M. Mubangizi, J. Quinn and N. Lawrence. “Monitoring Short Term Changes of Infectious Diseases in Uganda with Gaussian Processes”. In: *Advanced Analysis and Learning on Temporal Data: First ECML PKDD Workshop, AALTD 2015, Porto, Portugal, September 11, 2015, Revised Selected Papers*. Springer International Publishing, 2016, pp. 95–110.

[4] E. M. Constantinescu and M. Anitescu. “Physics-based covariance models for Gaussian processes with multiple outputs”. In: *International Journal for Uncertainty Quantification* 3.1 (2013), pp. 47–71.

[5] S. Da Veiga and A. Marrel. “Gaussian process modeling with inequality constraints”. eng. In: *Annales de la faculté des sciences de Toulouse Mathématiques* 21.3 (2012), pp. 529–555.

[6] R. Garnett. *Lecture 11: Bayesian Quadrature*. University Lecture. Accessed: 2019-05-01. Available from: `http://www.cse.wustl.edu/~garnett/cse515t/spring_2018/files/lecture_notes/11.pdf`. 2018.

[7] Z. Ghahramani. “Probabilistic machine learning and artificial intelligence”. In: *Nature* 521 (2015), pp. 452–459.

[8] D. Ginsbourger, O. Roustant and N. Durrande. “On degeneracy and invariances of random fields paths with applications in Gaussian process modelling”. In: *Journal of Statistical Planning and Inference* (2016), pp. 117–128.

[9] T. Graepel. “Solving Noisy Linear Operator Equations by Gaussian Processes: Application to Ordinary and Partial Differential Equations”. In: *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*. Aug. 2003.

[10] P. Hennig and M. Kiefel. “Quasi-Newton methods: A new direction”. In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 843–865.

[11] O. Koyejo, C. Lee and J. Ghosh. “Constrained Gaussian Process Regression for Gene-Disease Association”. In: *Proceedings of the IEEE 13th International Conference on Data Mining Workshops* 00 (2013), pp. 72–79.

[12] D. G. Luenberger. *Optimization by vector space methods.* John Wiley & Sons, Inc, 1969.

[13] H. Maatouk and X. Bay. “Gaussian Process Emulators for Computer Experiments with Inequality Constraints”. In: *Mathematical Geosciences* (2017), pp. 1–26.

[14] A. K. W. Navarro, J. Frellsen and R. E. Turner. *The Multivariate Generalised von Mises Distribution: Inference and Applications.* Tech. rep. arXiv:1602.05003, Feb. 2016.

[15] N. C. Nguyen and J. Peraire. “Gaussian functional regression for linear partial differential equations”. In: *Computer Methods in Applied Mechanics and Engineering* 287 (2015), pp. 69–89.

[16] N. C. Nguyen and J. Peraire. “Gaussian functional regression for output prediction: Model assimilation and experimental design”. In: *Journal of Computational Physics* 309 (2016), pp. 52–68.

[17] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes.* New York: McGraw-Hill Education, 1991.

[18] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning.* MIT press, Cambridge, MA, 2006.

[19] J. Ross and J. Dy. “Nonparametric Mixture of Gaussian Processes with Constraints”. In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13).* Vol. 28. JMLR Workshop and Conference Proceedings, 2013, pp. 1346–1354.

[20] O. Rudovic and M. Pantic. “Shape-constrained Gaussian process regression for facial-point-based head-pose normalization””. In: *Proceedings of the International Conference on Computer Vision (ICCV).* 2011.

[21] M. Salzmann and R. Urtasun. “Implicitly Constrained Gaussian Process Regression for Monocular Non-Rigid Pose Estimation”. In: *Neural Information Processing Systems (NIPS).* 2010.

[22] S. Särkkä. “Linear Operators and Stochastic Partial Differential Equations in Gaussian Process Regression”. In: *Proceedings of the Artificial Neural Networks and Machine Learning – ICANN 2011.* Springer, 2011, pp. 151–158.

[23] C. Tran, V. Pavlovic and R. Kopp. *Gaussian Process for Noisy Inputs with Ordering Constraints*. Tech. rep. arXiv:1507.00052, July 2015.

[24] N. Wahlström. “Modeling of Magnetic Fields and Extended Objects for Localization Applications”. PhD thesis. Division of Automatic Control, Linköping University, 2015.

Paper II

Title

Probabilistic modelling and reconstruction of strain.

Authors

Carl Jidling, Johannes Hendriks, Niklas Wahlström, Alexander Gregg, Thomas B. Schön, Christopher Wensrich, and Adrian Wills.

Statement of contribution

The theoretical construction described in this paper emerged from discussions involving all authors with contributions well reflected in the author ordering. The implementation and writing was done by Carl Jidling mainly, although with important contributions from the remaining authors, especially regarding the physics descriptions and related work in that field.

Edited version of

C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. "Probabilistic modelling and reconstruction of strain". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

Probabilistic modelling and reconstruction of strain

Abstract

This paper deals with modelling and reconstruction of strain fields, relying upon data generated from neutron Bragg-edge measurements. We propose a probabilistic approach in which the strain field is modelled as a Gaussian process, assigned a covariance structure customised by incorporation of the so-called equilibrium constraints. The computational complexity is significantly reduced by utilising an approximation scheme well suited for the problem. We illustrate the method on simulations and real data. The results indicate a high potential and can hopefully inspire the concept of probabilistic modelling to be used within other tomographic applications as well.

1 Introduction

The goal of tomographic reconstruction is to build a map of an unknown quantity within an object using information gained from irradiation experiments. A well known example of this is X-ray imaging, where the unknown quantity might be, for instance, the bone density inside a human body.

Each measurement provides information about the amount of intensity that the ray has lost when passing through the material. Of course, a single measurement does not uniquely define the interior. However, processing a large number of measurements taken from many different angles allows for an accurate reconstruction of the internal structure.

While techniques such as X-ray imaging and MRI are concerned with scalar fields, we are in this work considering the reconstruction of the strain field – a second order tensor – within a deformed material. This is a significantly harder problem as it is a multidimensional quantity at each point. For simplicity, we are restricting the analysis to a planar problem, but the extension to three dimensions follows the same procedure.

The development of accurate strain measuring techniques is motivated by applications within several fields. One field with perhaps especially exciting application potential is *additive manufacturing*, which involves printing of three-dimensional metal structures. For instance, this is of interest for developers of fuel nozzles [28] and turbine blades [34] within the aerospace industry.

Several techniques enabling high-precision measurement of residual strain have been proposed in previous work. These are characterised as destructive, semi-destructive or non-destructive, where examples from each category includes slitting [15], ring-coring [26] and diffraction [3, 13], respectively.

Bragg-edge analysis [19, 20] is an alternative transmission-based approach aiming at reconstructing the entire three-dimensional strain-field. This is an important difference to the established techniques outlined above, including the diffraction-based strain tomography such as synchrotron X-ray measurements [9, 10]. The term Bragg-edge refers to rapid changes in the relative transmission rate, which are determined from Bragg's law and hence directly related to the wavelength [20].

Methods relying on the Bragg-edge idea have seen a significant progress during recent years, and provides an essential foundation for generation of high-resolution strain-images within polycrystalline materials [19, 29, 30, 37].

A practical method based on the Brag-edge technique has been proposed by [36]. This relies upon the assumption that the strain field is compatible (see Section D), which allows the measurement equation to be expressed in terms of the boundary displacements. The reconstruction is obtained by solving a least squares problem and providing the result as a boundary condition to a finite element solver. An extension of this method is given by [35] with application to a real-world problem presented in [6].

The solution technique presented in this work relies upon the concept of *probabilistic modelling* [5]. Probabilistic modelling refers to methods that are employing probability theory to encode uncertainties present in the problem and where the solution is obtained through statistical inference. The idea is based on the assumption that uncertainties are always present, mainly due to a limited amount of data and the presence of measurement noise. A natural way of encoding these uncertainties in the model is therefore to assign a probabilistic measure to the unknown quantities themselves.

The contribution of this paper is a new way of modelling and reconstructing strain fields from data generated by neutron Bragg-edge measurements. We are using a tailored Gaussian process (GP) [17] to model the strain field, and by utilising the fact that GPs are closed under linear transformations, the reconstruction of the strain field is obtained through GP regression. The model is customised by designing the associated covariance function with respect to the so-called equilibrium constraints, which guarantees a physical solution.

2 Problem formulation

Given a set of measurements generated from a neutron Bragg-edge experiment, the problem faced in this work is to reconstruct the strain at various positions within a *sample*. The sample is an object, in which we want to reconstruct strain. The sample is considered to be two-dimensional. The strain in such a sample can be represented using a symmetric 2 × 2-matrix $\boldsymbol{\epsilon}$ called the *strain tensor*. Any point in the sample has an assigned strain tensor. This assignment is described by the *strain field* $\boldsymbol{\epsilon}(\mathbf{x})$, which is a function mapping any point in space $\mathbf{x} = [x,\ y]^\mathsf{T} \in \mathbb{R}^2$ to a strain tensor $\boldsymbol{\epsilon}$. The strain field can be construed as

$$\boldsymbol{\epsilon}(\mathbf{x}) = \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) & \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{yx}(\mathbf{x}) & \epsilon_{yy}(\mathbf{x}) \end{bmatrix} \in \mathbb{R}^{2\times 2}, \tag{1}$$

where $\epsilon_{xy}(\mathbf{x}) = \epsilon_{yx}(\mathbf{x})$ since strain tensors are symmetric.

The experiments rely upon high resolution time-of-flight neutron detectors. Neutron beams are generated at a source, transmitted through the sample, and recorded at a detector located at the opposite side of the sample. Considering a single measurement, assume that the neutrons enters the sample at a point $\mathbf{x}^0$, propagates along the direction defined by the unit vector $\hat{\mathbf{n}}$ and exits at $\mathbf{x}^0 + L\hat{\mathbf{n}}$, where L is the illuminated distance in the sample. This is illustrated in Fig. 1. An ideal measurement obtained from the neutron Bragg-edge method can be expressed in terms of the *Longitudinal Ray Transform* (LRT)

$$\mathrm{I}(\boldsymbol{\eta}) = \frac{1}{L}\int_0^L \hat{\mathbf{n}}^\mathsf{T}\boldsymbol{\epsilon}(\mathbf{x}^0 + s\hat{\mathbf{n}})\hat{\mathbf{n}}ds, \tag{2}$$

where $\boldsymbol{\eta} = \{\mathbf{x}^0, L, \hat{\mathbf{n}}\}$ specifies the argument of the LRT and where s is a coordinate used to specify the position on the line between the entry and exit points. We can interpret (2) as the average strain along the propagated path, so the LRT plays an important role in defining an adequate measurement model within this framework [11]. See A for some more details on the Bragg-edge experiment.

In a real-world application, we will have access to a data set $\mathcal{D} = \{(\boldsymbol{\eta}_i, y_i)\}_{i=1}^N$ with

$$y_i = \mathrm{I}(\boldsymbol{\eta}_i) + \varepsilon_i, \tag{3}$$

where ε_i denotes the measurement noise. Hence, the problem to be solved is this: Given the measurements $\{y_i\}_{i=1}^N$ in (3) and the relation (2), reconstruct the strain tensor (1) in any arbitrary *test point* $\mathbf{x}_*$ along with an uncertainty measure of the accuracy for this reconstruction.

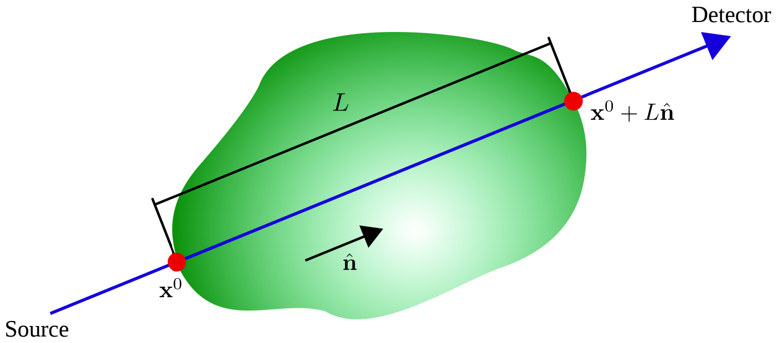


Figure 1: Experimental setup. The neutrons are transmitted from a source through the sample along the line starting at $\mathbf{x}^0$ and ending at $\mathbf{x}^0 + \hat{\mathbf{n}}L$ and finally measured by a detector.

3 Strain field reconstruction using Gaussian processes

In this work, the problem is solved by making use of the framework known as *probabilistic modelling*, with the somehow abstract idea of assigning a probability distribution to the strain field. This does not mean that we should think of the strain field as a random object, but the distribution provides a way for us to encode the uncertainty of the reconstruction. The presence of the uncertainty is mainly due to the fact that we have a finite number of measurements and that each of these measurements by construction are contaminated with noise.

More specifically, to reconstruct the strain field based on LRT measurements (2), we need a probabilistic model describing (i) the strain field, and (ii) the relation between the strain field and the measurements. The model should be able to reason about the uncertainties present in the measurements and take known physical constraints into consideration. In this work we choose to model the strain field with a Gaussian process.

3.1 Gaussian processes

A GP is a stochastic process suitable for modelling spatially correlated measurements. GPs can be seen as a distribution over functions

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}\big(\mathbf{m}(\mathbf{x}), \mathbf{K}(\mathbf{x}, \mathbf{x}')\big), \tag{4}$$

where the process is uniquely defined with its mean function $\mathbf{m}(\mathbf{x}) = \mathbb{E}[\mathbf{f}(\mathbf{x})]$ and covariance function $\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathbb{E}\big[(\mathbf{f}(\mathbf{x}) - \mathbf{m}(\mathbf{x}))(\mathbf{f}(\mathbf{x}') - \mathbf{m}(\mathbf{x}'))^\mathsf{T}\big]$. The GP is a generalisation of the multivariate Gaussian probability distribution in the sense that the function values evaluated for a finite number of inputs $\mathbf{x}_1, \dots, \mathbf{x}_N$ are Gaussian distributed

$$\begin{bmatrix} \mathbf{f}(\mathbf{x}_1) \\ \vdots \\ \mathbf{f}(\mathbf{x}_N) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}), \quad \text{where} \quad \boldsymbol{\mu} = \begin{bmatrix} \mathbf{m}(\mathbf{x}_1) \\ \vdots \\ \mathbf{m}(\mathbf{x}_N) \end{bmatrix}, \tag{5a}$$

and

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}(\mathbf{x}_1, \mathbf{x}_1) & \cdots & \mathbf{K}(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & & \vdots \\ \mathbf{K}(\mathbf{x}_N, \mathbf{x}_1) & \cdots & \mathbf{K}(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}. \tag{5b}$$

In this work we will only consider zero-mean GPs, i.e., where $\mathbf{m}(\mathbf{x}) = \mathbf{0}$. This is the natural choice in absence of more specific prior knowledge. It shall not be interpreted as if we believe that $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, it simply represents the fact that we do not have any better initial guess.

Since $\boldsymbol{\epsilon}(\mathbf{x})$ is a symmetric 2×2 tensor, it consists of three unknown components, $\epsilon_{xx}(\mathbf{x})$, $\epsilon_{xy}(\mathbf{x})$ and $\epsilon_{yy}(\mathbf{x})$. We therefore choose to model the strain tensor with a function $\mathbf{f}(\mathbf{x}) : \mathbb{R}^2 \mapsto \mathbb{R}^3$, here called the *strain function*, where

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_{xx}(\mathbf{x}) \\ f_{xy}(\mathbf{x}) \\ f_{yy}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) \\ \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{yy}(\mathbf{x}) \end{bmatrix} \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}. \tag{6}$$

We then put a GP prior on $\mathbf{f}(\mathbf{x})$ according to (4). The model now consist of two parts:

1. The GP prior (4) of the strain field, i.e., our choice of $\mathbf{K}(\mathbf{x}, \mathbf{x}')$. This GP prior is described in Section 3.2.

2. The joint distribution between the measurements $\mathbf{y} = [y_1,\, y_2,\, \dots, y_N]^\mathsf{T}$ and the strain function $\mathbf{f}_* = \mathbf{f}(\mathbf{x}_*)$ at a point $\mathbf{x}_*$ where we want to make a reconstruction. This distribution allows us to infer the measurements and it is described in Section 3.3 and Section 3.4.

3.2 The covariance function

The strain function cannot be any arbitrary function mapping $\mathbb{R}^2$ to $\mathbb{R}^3$. It needs to obey some physical laws. Therefore, we want to model the covariance function $\mathbf{K}(\mathbf{x}, \mathbf{x}')$ such that any sample we draw from the GP prior (4) is a valid strain function. More specifically we need to fulfil the equilibrium constraints, which for isotropic linearly elastic (i.e. in the absence of strong grain texture) solid materials under the assumption of plane stress, read as

$$\frac{\partial f_{xx}(\mathbf{x})}{\partial x} + (1-\nu)\frac{\partial f_{xy}(\mathbf{x})}{\partial y} + \nu\frac{\partial f_{yy}(\mathbf{x})}{\partial x} = 0, \tag{7a}$$

$$\nu\frac{\partial f_{xx}(\mathbf{x})}{\partial y} + (1-\nu)\frac{\partial f_{xy}(\mathbf{x})}{\partial x} + \frac{\partial f_{yy}(\mathbf{x})}{\partial y} = 0, \tag{7b}$$

where ν denotes Poisson's ratio. For $\mathbf{f}(\mathbf{x})$ to fulfill these constraints we can describe it as a transformation of another scalar function $\varphi(\mathbf{x}) : \mathbb{R}^3 \mapsto \mathbb{R}$ via the transfomation according to

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2}{\partial y^2} - \nu\frac{\partial^2}{\partial x^2} \\ -(1+\nu)\frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial x^2} - \nu\frac{\partial^2}{\partial y^2} \end{bmatrix} \varphi(\mathbf{x}) = \mathcal{L}_{\mathbf{x}}\varphi. \tag{8}$$

Here, $\mathcal{L}_{\mathbf{x}}$ is an *operator* mapping scalar potential functions $\varphi(\mathbf{x})$ to vector-valued strain functions $\mathbf{f}(\mathbf{x})$, i.e., in this case we have $\mathcal{L}_{\mathbf{x}} : (\mathbb{R}^2 \mapsto \mathbb{R}) \mapsto (\mathbb{R}^2 \mapsto \mathbb{R}^3)$. One can easily verify that the strain function in (8) fulfils the constraints in (7). This transformation is derived from the underlying physics where $\varphi(\mathbf{x})$ is a known physical potential called the *Airy stress function*, for details, see C. We can also derive the transformation $\mathbf{f}(\mathbf{x}) = \mathcal{L}_{\mathbf{x}}\varphi$ from any set of linear constraints, for example those presented in (7), following the procedure described by [8].

Instead of designing a covariance function for the strain function $\mathbf{f}(\mathbf{x})$, we design a GP prior for the scalar potential function $\varphi(\mathbf{x})$

$$\varphi(\mathbf{x}) \sim \mathcal{GP}\big(0, k_\varphi(\mathbf{x}, \mathbf{x}')\big). \tag{9}$$

It can be easily verified in (8) that the operator $\mathcal{L}_{\mathbf{x}}$ is linear

$$\mathcal{L}_{\mathbf{x}}[\gamma_1\varphi_1(\mathbf{x}) + \gamma_2\varphi_2(\mathbf{x})] = \gamma_1\mathcal{L}_{\mathbf{x}}[\varphi_1(\mathbf{x})] + \gamma_2\mathcal{L}_{\mathbf{x}}[\varphi_2(\mathbf{x})].$$

Since the GP $\varphi(\mathbf{x})$ is mapped through this linear operator, it follows from B that $\mathbf{f}(\mathbf{x})$ is also a GP

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}(\mathbf{x}, \mathbf{x}')), \tag{10}$$

where

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathcal{L}_{\mathbf{x}} k_\varphi(\mathbf{x}, \mathbf{x}') \mathcal{L}_{\mathbf{x}'}^{\mathsf{T}}. \tag{11}$$

Any sample drawn from the GP prior (10) will by this design obey the equilibrium constraints (7) and hence represent a valid strain field. There are a variety of options for the scalar covariance function k_φ. The most common one is the so-called *squared exponential* covariance function

$$k_\varphi(\mathbf{x}, \mathbf{x'}) = \sigma_f^2 \exp\left[-\frac{1}{2}(l_x^{-2}r_x^2 + l_y^{-2}r_y^2)\right], \tag{12}$$

where $r_x = x - x'$ and $r_y = y - y'$. Here, σ_f is a magnitude parameter while l_x and l_y determine the rate at which the covariance decays in direction x and y, respectively. These so-called *hyperparameters* are learnt from data, more on this in Section 4.2.

Note that we can write

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathcal{L}_\mathbf{x} k_\varphi(\mathbf{x}, \mathbf{x}')\mathcal{L}_{\mathbf{x}'}^\mathsf{T} = \mathcal{L}_\mathbf{x}\mathcal{L}_{\mathbf{x}'}^\mathsf{T} k_\varphi(\mathbf{x}, \mathbf{x}') = \mathbf{\Psi} k_\varphi(\mathbf{x}, \mathbf{x}'),$$

where $\mathbf{\Psi}$ is a matrix of operators. Specifically,

$$\mathbf{\Psi}_{11} = \nu^2 \frac{\partial^4}{\partial x^2 \partial x'^2} - 2\nu \frac{\partial^4}{\partial x^2 \partial y'^2} + \frac{\partial^4}{\partial y^2 \partial y'^2}, \tag{13a}$$

$$\mathbf{\Psi}_{22} = (\nu + 1)^2 \frac{\partial^4}{\partial x \partial y \partial x' \partial y'}, \tag{13b}$$

$$\mathbf{\Psi}_{33} = \frac{\partial^4}{\partial x^2 \partial x'^2} - 2\nu \frac{\partial^4}{\partial x^2 \partial y'^2} + \nu^2 \frac{\partial^4}{\partial y^2 \partial y'^2}, \tag{13c}$$

$$\mathbf{\Psi}_{12} = \mathbf{\Psi}_{21} = -\nu(\nu + 1) \frac{\partial^4}{\partial x \partial y \partial x'^2} + (\nu + 1) \frac{\partial^4}{\partial x \partial y \partial y'^2}, \tag{13d}$$

$$\mathbf{\Psi}_{13} = \mathbf{\Psi}_{31} = -\nu \frac{\partial^4}{\partial x^2 \partial x'^2} + (\nu^2 + 1) \frac{\partial^4}{\partial x^2 \partial y'^2} - \nu \frac{a}{b} \frac{\partial^4}{\partial y^2 \partial y'^2}, \tag{13e}$$

$$\mathbf{\Psi}_{23} = \mathbf{\Psi}_{32} = -\nu(\nu + 1) \frac{\partial^4}{\partial x \partial y \partial y'^2} + (\nu + 1) \frac{\partial^4}{\partial x \partial y \partial x'^2}. \tag{13f}$$

For example, if we let $k_\varphi(\mathbf{x}, \mathbf{x}')$ be the squared exponential covariance function (12) we get

$$\frac{\partial^4}{\partial x^2 \partial x'^2} k_\varphi = l_x^{-4}(l_x^{-4}r_x^4 - 6l_x^{-2}r_x^2 + 3)k_\varphi, \tag{14a}$$

$$\frac{\partial^4}{\partial x^2 \partial y'^2} k_\varphi = \frac{\partial^4}{\partial x \partial y \partial x' \partial y'} k_\varphi l_x^{-2} l_y^{-2}(1 - l_x^{-2}r_x^2)(1 - l_y^{-2}r_y^2)k_\varphi, \tag{14b}$$

$$\frac{\partial^4}{\partial x \partial y \partial y'^2} k_\varphi = l_x^{-2} l_y^{-4} r_x r_y (l_x^{-2} r_y^2 - 3)k_\varphi, \tag{14c}$$

with the remaining expressions obtained by exchanging $x \leftrightarrow y$.

3.3 The measurement model

In the previous subsection we presented a GP model for the strain function stipulating that the strain tensors at any two points will be jointly Gaussian distributed. Further, the measurement model (2) defines a relationship between the strain function and the measurements. We will use this relation to define a joint distribution between the two, which later will be used to do the inference.

First we reformulate the integrator of (2) as

$$\hat{\mathbf{n}}^\mathsf{T}\boldsymbol{\epsilon}(\mathbf{x})\hat{\mathbf{n}} = \begin{bmatrix} n_x & n_y \end{bmatrix} \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) & \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{yx}(\mathbf{x}) & \epsilon_{yy}(\mathbf{x}) \end{bmatrix} \begin{bmatrix} n_x \\ n_y \end{bmatrix}$$
$$= \underbrace{\begin{bmatrix} n_x^2 & 2n_x n_y & n_y^2 \end{bmatrix}}_{\triangleq \vec{\mathbf{n}}^\mathsf{T}} \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) \\ \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{yy}(\mathbf{x}) \end{bmatrix} = \vec{\mathbf{n}}^\mathsf{T}\mathbf{f}(\mathbf{x}), \quad (15)$$

such that

$$\mathrm{I}(\boldsymbol{\eta}) = \boldsymbol{\vartheta}_{\boldsymbol{\eta}}[\mathbf{f}] = \frac{1}{L}\int_0^L \vec{\mathbf{n}}^\mathsf{T}\mathbf{f}(\mathbf{x}^0 + s\hat{\mathbf{n}})ds. \quad (16)$$

Here, $\boldsymbol{\vartheta}_{\boldsymbol{\eta}}$ is also considered to be an operator that maps strain functions $\mathbf{f}(\mathbf{x})$ into LRT functions $\mathrm{I}(\boldsymbol{\eta})$. This operator is also linear and as a consequence, the Gaussianity will be preserved also for the joint distribution of $\mathbf{f}_* = \mathbf{f}(\mathbf{x}_*)$ and $\mathbf{y} = [y_1, y_2 \dots, y_N]^\mathsf{T}$. We denote this joint Gaussian distribution as

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}_\mathrm{I} + \sigma^2 I & \mathbf{K}_* \\ \mathbf{K}_*^\mathsf{T} & \mathbf{K}_{**} \end{bmatrix} \right), \quad (17)$$

where $\mathbf{K}_{**}$ denotes the covariance of $\mathbf{f}_*$, $\mathbf{K}_\mathrm{I} + \sigma^2 I$ denotes the covariance of $\mathbf{y}$ and $\mathbf{K}_*$ denotes the cross-covariance between $\mathbf{y}$ and $\mathbf{f}_*$. The covariance of $\mathbf{f}_*$ is provided by the covariance function from the GP prior

$$\mathbf{K}_{**} = \mathbb{E}\left[\mathbf{f}(\mathbf{x}_*)\mathbf{f}(\mathbf{x}_*)^\mathsf{T}\right] = \mathbf{K}(\mathbf{x}_*, \mathbf{x}_*), \quad (18)$$

where we use the fact that $\mathbb{E}[\mathbf{f}(\mathbf{x})] = 0$. The cross-covariance between an LRT measurement y_i and the strain function $\mathbf{f}_*$ can be computed based on (16) as

$$(\mathbf{K}_*)_i = \mathbb{E}\left[\mathrm{I}(\boldsymbol{\eta}_i)\mathbf{f}(\mathbf{x}_*)^\mathsf{T}\right] = \frac{1}{L_i}\int_0^{L_i} \vec{\mathbf{n}}_i^\mathsf{T}\mathbb{E}\left[\mathbf{f}(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i)\mathbf{f}(\mathbf{x}_*)^\mathsf{T}\right] ds$$
$$= \frac{1}{L_i}\int_0^{L_i} \vec{\mathbf{n}}_i^\mathsf{T}\mathbf{K}(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i, \mathbf{x}_*)ds, \quad (19a)$$

where $(\mathbf{K}_*)_i$ denotes the ith row in the matrix $\mathbf{K}_*$ and where we also use the fact that $\mathbb{E}[y_i] = \mathbb{E}[\mathrm{I}(\boldsymbol{\eta}_i)] + \mathbb{E}[\varepsilon_i] = 0$. In a similar manner, we can also compute the covariance of the measurements $\mathbb{E}[y_i y_j] = (\mathbf{K}_\mathrm{I})_{ij} + \sigma^2 \delta_{ij}$, where

$$\begin{aligned}(\mathbf{K}_\mathrm{I})_{ij} &= \mathbb{E}[\mathrm{I}(\boldsymbol{\eta}_i)\mathrm{I}(\boldsymbol{\eta}_j)] \\ &= \frac{1}{L_i L_j} \int_0^{L_j} \int_0^{L_i} \vec{\mathbf{n}}_i^\mathsf{T} \mathbf{K}(\mathbf{x}_i^0 + s_i \hat{\mathbf{n}}_i, \mathbf{x}_j^0 + s_j \hat{\mathbf{n}}_j) \vec{\mathbf{n}}_j ds_i ds_j. \quad (19b)\end{aligned}$$

To specify the full joint covariance in (17), the integrals in (19) can not be expected to have an analytical solution. However, numerical integration can be avoided for instance by making use of the approximation technique described in Section 2.6.

Based on the joint distribution (17) we can condition the strain function $\mathbf{f}_* = \mathbf{f}(\mathbf{x}_*)$ on the measurements $\mathbf{y}$ to get a posterior. Due to the Gaussianity and the linear operation of conditioning, also this posterior will be Gaussian distributed according to

$$\mathbf{f}_*|\mathbf{y} \sim \mathcal{N}\Big(\boldsymbol{\mu}_{\mathbf{f}_*|\mathbf{y}}, \mathbf{K}_{\mathbf{f}_*|\mathbf{y}}\Big), \quad (20a)$$

where

$$\boldsymbol{\mu}_{\mathbf{f}_*|\mathbf{y}} = \mathbf{K}_*^\mathsf{T}(\mathbf{K}_\mathrm{I} + \sigma^2 \mathbf{I})^{-1}\mathbf{y}, \quad (20b)$$

$$\mathbf{K}_{\mathbf{f}_*|\mathbf{y}} = \mathbf{K}_{**} - \mathbf{K}_*^\mathsf{T}(\mathbf{K}_\mathrm{I} + \sigma^2 \mathbf{I})^{-1}\mathbf{K}_*. \quad (20c)$$

The mean $\boldsymbol{\mu}_{\mathbf{f}_*|\mathbf{y}}$ is the reconstructed strain function at position $\mathbf{x}_*$ and its associated covariance matrix $\mathbf{K}_{\mathbf{f}_*|\mathbf{y}}$ encodes the uncertainty of this reconstruction.

The extension to multiple test points is straightforward. We then need to modify the matrices $\mathbf{K}_*$ and $\mathbf{K}_{**}$. Consider the set of M test points $\{\mathbf{x}_*^j\}_{j=1}^M$. The dimension of $\mathbf{K}_*$ will change from $N \times 3$ to $N \times 3M$, where the columns $3j - 2$ to $3j$ is built up according to (19a) with $\mathbf{x}_* = \mathbf{x}_*^j$. As for $\mathbf{K}_{**}$, this matrix dimension will change from 3×3 to $3M \times 3M$, encoding the covariance between all test points. Hence, it will be built up by $M \times M$ blocks each of size 3×3, with block (i, j) being $\mathbf{K}(\mathbf{x}_*^i, \mathbf{x}_*^j)$. The conditioning (9) is then performed in the same way. This set of test points can for example be a fairly dense grid covering the whole region of interest where we want to do the reconstruction.

3.4 Extension of measurement model

So far we have restricted the model to the case where each neutron beam passes through the sample only once on its way from the source to the detector. In the general case, however, we must allow the beam to pass through several segments of the sample. We denote the starting points of

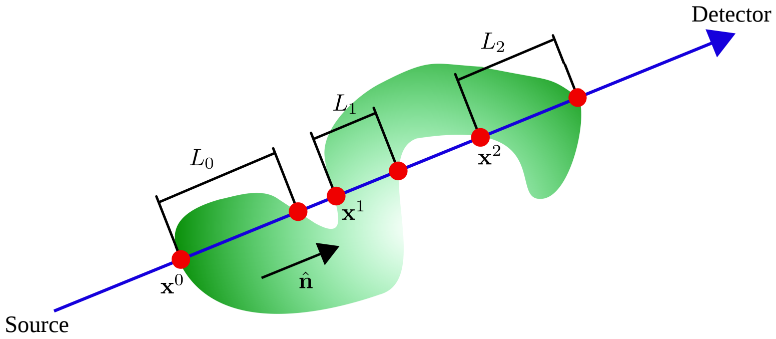


Figure 2: Illustration of the case where the neutron beam passes through $P = 3$ segments of the sample.

the P different segments with $\mathbf{x}^k$ and the corresponding end points with $\mathbf{x}^k + L^k\hat{\mathbf{n}}$ as illustrated in Fig. 2. The measurement is still to be interpreted as the average strain tensor along these line segments. Therefore, we have to integrate along all of these line segments and normalise with the total length $\sum_{k=0}^{P-1} L^k$ travelled though the sample. The measurement equation (16) then turns into

$$\mathrm{I}(\boldsymbol{\eta}) = \boldsymbol{\vartheta}_{\boldsymbol{\eta}}[\mathbf{f}] = \frac{1}{\sum_{k=0}^{P-1} L^k} \sum_{k=0}^{P-1} \int_0^{L^k} \vec{\mathbf{n}}^\mathsf{T} \mathbf{f}(\mathbf{x}^k + s\hat{\mathbf{n}}) ds. \tag{21}$$

Here, $\boldsymbol{\eta}$ consists of all arguments for all segments

$$\boldsymbol{\eta} = \{\mathbf{x}^0, L^0, \ldots, \mathbf{x}^{P-1}, L^{P-1}, \hat{\mathbf{n}}\}.$$

Note that the direction $\hat{\mathbf{n}}$ is the same for all segments. Although this modification requires some care in the implementation, there is no conceptual challenge added. The covariance matrices in (19) will change accordingly

$$(\mathbf{K}_*)_i = \frac{1}{\sum_{k=0}^{P_i-1} L_i^k} \sum_{k=0}^{P_i-1} \int_0^{L_i^k} \vec{\mathbf{n}}_i^\mathsf{T} \mathbf{K}(\mathbf{x}_i^k + s\hat{\mathbf{n}}_i, \mathbf{x}_*) ds, \tag{22a}$$

$$(\mathbf{K}_\mathrm{I})_{ij} = \frac{1}{\left(\sum_{k=0}^{P_i-1} L_i^k\right)\left(\sum_{l=0}^{P_j-1} L_j^l\right)} \cdot \\ \cdot \sum_{k=0}^{P_i-1} \sum_{l=0}^{P_j-1} \left[\int_0^{L_j^l} \int_0^{L_i^k} \vec{\mathbf{n}}_i^\mathsf{T} \mathbf{K}(\mathbf{x}_i^k + s_i\hat{\mathbf{n}}_i, \mathbf{x}_j^l + s_j\hat{\mathbf{n}}_j) \vec{\mathbf{n}}_j ds_i ds_j \right]. \tag{22b}$$

The reconstruction procedure described in (9) remains the same with these new covariance matrices in place.

4 The model in practice

Before presenting the numerical results, we discuss a couple of important practical aspects concerning the computational complexity and the hyperparameter selection. The approach as described below is summarised in Algorithm 1.

4.1 Reducing the computational complexity

A bottleneck in GP regression is the storage and inversion of the matrix $\mathbf{K}_{\mathrm{I}} + \sigma^2\mathbf{I}$ in (9), which scales as $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$ in memory and time, respectively. For large data sets, approximate methods are motivated by the need to decrease the problem size, making memory requirements manageable and reducing the runtime. There are a variety of methods described in the literature, and we refer to [16] for a useful review. Here we make use of the approximation method proposed by [25], which turns out to fit our model very well. A drawback with this method is that it requires the covariance function to be *stationary*, which means that it is dependent only on the difference between the input locations. However, that is true for some of the most common ones, including the squared exponential (12).

The key idea is to estimate the covariance function as a truncated sum of m basis functions

$$k_{\varphi}(\mathbf{x}, \mathbf{x}') \approx \sum_{i=1}^{m} S(\boldsymbol{\lambda}_i)\phi_i(\mathbf{x})\phi_i(\mathbf{x}'), \tag{23}$$

where S is the *spectral density* of the covariance function. For a stationary covariance function $k_{\varphi} = k_{\varphi}(\boldsymbol{r})$ where $\boldsymbol{r} = \mathbf{x} - \mathbf{x}'$, the spectral density is given by

$$S(\boldsymbol{\omega}) = \int k(\boldsymbol{r})e^{-\mathrm{i}\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{r}}d\boldsymbol{r}. \tag{24}$$

The basis functions $\phi_i(\mathbf{x})$ and eigenvalues $\boldsymbol{\lambda}_i$ are obtained from the solution to the Laplace eigenvalue problem on the domain Ω

$$\begin{cases} -\Delta\phi_i(\mathbf{x}) = \|\boldsymbol{\lambda}_i\|^2\phi_i(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \phi_i(\mathbf{x}) = 0, & \mathbf{x} \in \partial\Omega. \end{cases} \tag{25}$$

The Dirichlet boundary condition is the most natural choice, although any boundary condition could be chosen. For a two-dimensional domain $\Omega = [-\rho_x, \rho_x] \times [-\rho_y, \rho_y]$, the solution of (10) reads

$$\phi_i(\mathbf{x}) = \frac{1}{\sqrt{\rho_x\rho_y}} \sin\left(\lambda_{i_x}(x + \rho_x)\right) \sin\left(\lambda_{i_y}(y + \rho_y)\right), \tag{26a}$$

$$\lambda_{i_x} = \frac{\pi i_x}{2\rho_x}, \qquad \lambda_{i_y} = \frac{\pi i_y}{2\rho_y}, \tag{26b}$$

where i_x and i_y are chosen such that the eigenvalues lie in a desired frequency domain, and the size of Ω should be adjusted thereafter. Loosely speaking, the choice of ρ_x determines the frequency resolution in the x-direction, and similarly for y.

The approximate posterior expressions are

$$\mathbb{E}[f_*] \approx \boldsymbol{\phi}_*^\mathsf{T}(\boldsymbol{\Phi}\boldsymbol{\Phi}^\mathsf{T} + \sigma^2\boldsymbol{\Lambda}^{-1})^{-1}\boldsymbol{\Phi}\mathbf{y}, \tag{27a}$$

$$\mathbb{V}[f_*] \approx \sigma^2\boldsymbol{\phi}_*^\mathsf{T}(\boldsymbol{\Phi}\boldsymbol{\Phi}^\mathsf{T} + \sigma^2\boldsymbol{\Lambda}^{-1})^{-1}\boldsymbol{\phi}_*, \tag{27b}$$

where $\boldsymbol{\Phi}_{ij} = \phi_i(\mathbf{x}_j)$, $\boldsymbol{\phi}_* = [\phi_1(\mathbf{x}_*)\dots\phi_m(\mathbf{x}_*)]^\mathsf{T}$ and $\boldsymbol{\Lambda}_{jj} = S(\boldsymbol{\lambda}_j)$. The correct expressions for our problem are found by projecting the transformation given by (16) onto the basis functions. We end up with

$$\mathbb{E}[\mathbf{f}_*] \approx \mathbf{Q}_*^\mathsf{T}(\mathbf{Q}\mathbf{Q}^\mathsf{T} + \sigma^2\boldsymbol{\Lambda}^{-1})^{-1}\mathbf{Q}\mathbf{y}, \tag{28a}$$

$$\mathbb{V}[\mathbf{f}_*] \approx \sigma^2\mathbf{Q}_*^\mathsf{T}(\mathbf{Q}\mathbf{Q}^\mathsf{T} + \sigma^2\boldsymbol{\Lambda}^{-1})^{-1}\mathbf{Q}_*. \tag{28b}$$

where

$$\mathbf{Q}_* = \begin{bmatrix}\mathcal{L}_\mathbf{x}\phi_1|_{\mathbf{x}=\mathbf{x}_*} & \dots & \mathcal{L}_\mathbf{x}\phi_m|_{\mathbf{x}=\mathbf{x}_*}\end{bmatrix}^\mathsf{T}, \tag{29a}$$

$$\mathbf{Q}_{ij} = \boldsymbol{\vartheta}_{\boldsymbol{\eta}_j}[\mathcal{L}_\mathbf{x}\phi_i]. \tag{29b}$$

Comparing this with (22), we have used that $\mathbf{K}_* \approx \mathbf{Q}^\mathsf{T}\boldsymbol{\Lambda}\mathbf{Q}_*$ and $\mathbf{K}_\mathrm{I} \approx \mathbf{Q}^\mathsf{T}\boldsymbol{\Lambda}\mathbf{Q}$, and the computationally more preferable form (29) is obtained by utilising the identities

$$\mathbf{P}\mathbf{B}^\mathsf{T}(\mathbf{B}\mathbf{P}\mathbf{B}^\mathsf{T} + \mathbf{R})^{-1} = (\mathbf{B}^\mathsf{T}\mathbf{R}^{-1}\mathbf{B} + \mathbf{P}^{-1})^{-1}\mathbf{B}^\mathsf{T}\mathbf{R}^{-1},$$

and

$$\mathbf{A} - \mathbf{A}\mathbf{C}^\mathsf{T}(\mathbf{C}\mathbf{A}\mathbf{C}^\mathsf{T} + \mathbf{W})^{-1}\mathbf{C}\mathbf{A} = (\mathbf{C}^\mathsf{T}\mathbf{W}^{-1}\mathbf{C} + \mathbf{A}^{-1})^{-1}.$$

This approximation scheme reduces the complexity of the regression from $\mathcal{O}(N^3)$ to $\mathcal{O}(Nm^2)$. The actual savings in our case are even larger, since the hazard of numerically computing the integrals in (22) is removed: now all we need is to compute single integrals, and this is done analytically due to the simple form of the basis functions (for details see D). Hence, for this particular problem, the approximation is computationally preferable even if $N < m$.

Extending to multiple test points $\{\mathbf{x}_*^j\}_{j=1}^M$, all we need to change is $\mathbf{Q}_*$, so that (29a) becomes

$$\mathbf{Q}_* = \begin{bmatrix}\mathcal{L}_\mathbf{x}\phi_1|_{\mathbf{x}=\mathbf{x}_*^1} & \dots & \mathcal{L}_\mathbf{x}\phi_m|_{\mathbf{x}=\mathbf{x}_*^1} \\ \vdots & \vdots & \vdots \\ \mathcal{L}_\mathbf{x}\phi_1|_{\mathbf{x}=\mathbf{x}_*^M} & \dots & \mathcal{L}_\mathbf{x}\phi_m|_{\mathbf{x}=\mathbf{x}_*^M}\end{bmatrix}^\mathsf{T}. \tag{30}$$

Algorithm 1 Reconstructing the GP with the reduced-rank approach

```
Input: $\mathcal{D} = \{(\boldsymbol{\eta}_i, y_i)\}_{i=1}^N$, $\mathbf{x}_*$, $\Omega$, $m$.
Output: $\mathbb{E}[\mathbf{f}(\mathbf{x}_*)], \mathbb{V}[\mathbf{f}(\mathbf{x}_*)]$.
1: Construct the matrix $\mathbf{Q}$ as defined in (29b), with details given in D.
2: Optimise the hyperparameters $\boldsymbol{\theta}$ as described in Section 4.2.
3: Construct $\mathbf{Q}_*$ as defined in (29a).
4: Solve the GP regression problem by (28).
```

The expressions (28) are then applied as before. Note that, usually what is desired is the variance of the different component values in each test point, and not the covariance between them. Hence, we should not compute the entire matrix in (28b), but only its diagonal elements.

4.2 Hyperparameters

The covariance function k_φ is characterised by its hyperparameters $\boldsymbol{\theta} = \{\theta_k\}$. An example was given in equation (12). This set does also include the noise level σ. Usually they are selected by maximising the marginal likelihood $p(\mathbf{y}|\{\boldsymbol{\eta}_i\}, \boldsymbol{\theta})$, which is the probability of the data conditioned on the input locations and the hyperparameters. The idea is to choose the hyperparameters $\boldsymbol{\theta}_*$ that given the choice of covariance function are most likely to have generated the observed data. The marginal likelihood and its derivatives can be computed in closed form [17]. For convenience, the logarithm of the marginal likelihood is usually considered, and it is for our problem given by

$$\log p(\mathbf{y}|\{\boldsymbol{\eta}_i\}, \boldsymbol{\theta}) = -\frac{1}{2}\log\det(\mathbf{K}_\mathrm{I} + \sigma^2 I) - \frac{1}{2}\mathbf{y}^\mathsf{T}(\mathbf{K}_\mathrm{I} + \sigma^2 I)^{-1}\mathbf{y} - \frac{N}{2}\log 2\pi, \tag{31}$$

where $\mathbf{K}_\mathrm{I}$ is a function of $\boldsymbol{\theta}$. Since the constant term is irrelevant for optimisation purposes, we get

$$\boldsymbol{\theta}_* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\Big[-\frac{1}{2}\log\det(\mathbf{K}_\mathrm{I} + \sigma^2 I) - \frac{1}{2}\mathbf{y}^\mathsf{T}(\mathbf{K}_\mathrm{I} + \sigma^2 I)^{-1}\mathbf{y}\Big]. \tag{32}$$

An approximative version of (32) is obtained by replacing $\mathbf{K}_\mathrm{I}$ with $\mathbf{Q}^\mathsf{T}\boldsymbol{\Lambda}\mathbf{Q}$. The derivative expressions for this case are given in D.2. The optimisation can thereafter be carried out using any standard gradient-based method, such as for example the BFGS algorithm [12].

5 Experimental results

5.1 Simulated experiment – cantilevered rectangular plate

As a simple example illustrating the potential of the method, consider the problem presented in [35]. A cantilevered plate is subject to a vertical load at the right end, see Fig. 3. The approximate equations for the strain field components are

$$\epsilon_{xx} = \frac{P}{EI}(l - x)y, \tag{33a}$$

$$\epsilon_{xy} = -\frac{(1+\nu)P}{2EI}\left(\frac{h^2}{4} - y^2\right), \tag{33b}$$

$$\epsilon_{yy} = -\frac{\nu P}{EI}(l - x)y, \tag{33c}$$

where $I = th^3/12$. Here, I denotes the moment of inertia, l, h, and t denotes the width, height and thickness of the plate, P denotes the magnitude of the load and E and ν denote Youngs modulus and Poisson's ratio, respectively. We are using the same numerical values as in [35], namely $E = 200\,\text{GPa}$, $\nu = 0.3$, $h = 10\,\text{mm}$, $t = 6\,\text{mm}$, $l = 20\,\text{mm}$ and $P = 2\,\text{kN}$. The standard deviation of the synthetic measurement noise is here $\sigma = 10^{-6}$, which was found suitable for illustration purpose. The covariance function used was constructed as described in Section 3.2 with $k_\varphi(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2)$.

The ϵ_{xx}-component according to (33a) is illustrated in Fig. 4a together with the paths along which the line integral (16) have been generated. The remaining three figures in Fig. 4 show the reconstructed component and its standard deviation using 1, 5 and 10 measurements, respectively.

The reconstruction was made by building the matrices defined in (19), and then applying (20b). It is interesting to note that only 10 measurements are enough for the method to produce a reconstruction that is very hard to visually distinguish from the true function. Note that the uncertainty is higher in regions further away from the measurements.

For comparison, we are performing a convergence study similar to the one performed by [35]. Here, the measured data is not generated using (33) but instead from a finite element solution of the problem, which should better reflect reality. To line up with the real experimental setting, the data is not randomly chosen but comes in chunks corresponding to different *projections*. Each projection typically contains a couple of hundred measurements taken from the same angle but uniformly distributed along the perpendicular direction.

We consider the synthetic noise level $\sigma = 10^{-4}$, to align with [35] where that value was used as an expected experimental measurement noise level. To

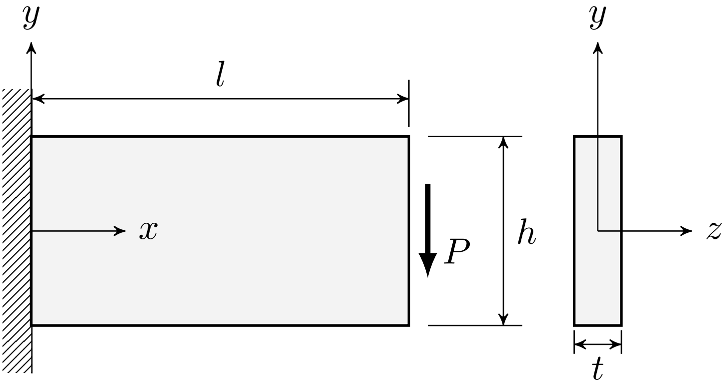


Figure 3: Rectangular plate of width l, height h and thickness t, cantilevered on the left side and subject to a vertical load P on the right. The approximate strain components within this plate are given by (33).

model the Airy stress function, we are here using the Matérn$_{5/2}$ covariance function

$$k_\varphi(\mathbf{x},\mathbf{x}') = \sigma_f^2\left(1+\frac{\sqrt{5}r}{l}+\frac{5r^2}{3l^2}\right)e^{-\frac{\sqrt{5}r}{l}}, \tag{34a}$$

$$r = \|\mathbf{x}-\mathbf{x}'\|^2, \tag{34b}$$

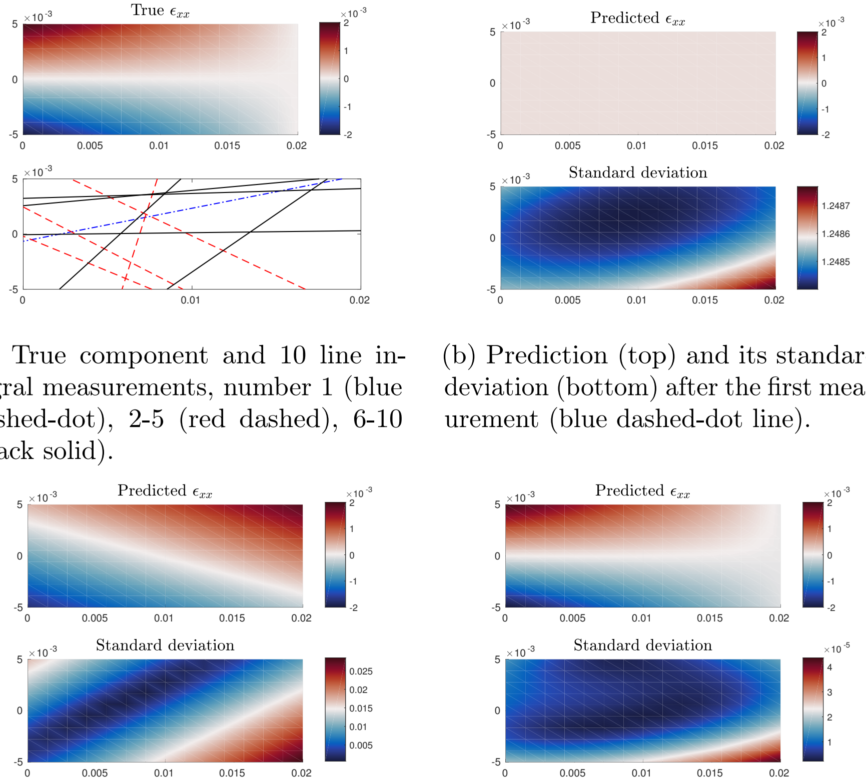
with the hyperparameters chosen by maximising the marginal likelihood (32). This covariance function belongs to a generalisation that relaxes the extreme smoothness assumptions of the squared exponential covariance function (12), and is often considered to be somewhat more realistic. The prediction is made in uniformly distributed points on a 40 × 20 mesh.

We are reporting the relative error in the reconstruction, where this involves a concatenation of all components in all points. For a total set of N projections, the angle from which projection k is taken has been chosen as $\frac{\pi}{96}+\lceil 95\frac{k-1}{N-1}-0.5\rceil\frac{\pi}{96}$, so the projection angles are approximately evenly spaced over $[0,\ \pi]$. Gaussian noise has been added to the measurements with Matlab:s default random seed.

We have here used the approximative method described in Section 2.6 with $\rho_x = 3l$, $\rho_y = 3h/2$ and a total number of 160 basis functions. The spectral density of the Matérn$_{5/2}$ covariance function (34a) is given by

$$S(\boldsymbol{\omega}) = \sigma_f^2 100\sqrt{5}\pi l^{-5}\left(\frac{5}{l^2}+\|\boldsymbol{\omega}\|^2\right)^{-7/2}\frac{\Gamma(7/2)}{\Gamma(5/2)}, \tag{35}$$

where $\Gamma(\cdot)$ is the gamma function.



(a) True component and 10 line integral measurements, number 1 (blue dashed-dot), 2-5 (red dashed), 6-10 (black solid).

(b) Prediction (top) and its standard deviation (bottom) after the first measurement (blue dashed-dot line).

(c) Prediction (top) and its standard deviation (bottom) after 5 measurements (blue dashed-dot line and red dashed lines).

(d) Prediction (top) and its standard deviation (bottom) after all measurements.

Figure 4: True and reconstructed strain field including its standard deviation for the ϵ_{xx}-component in the rectangular plate shown in Fig. 3. The measurements are the line integral paths shown in the top left figure. Note that the uncertainty is larger at the regions that are poorly covered by the measurements.

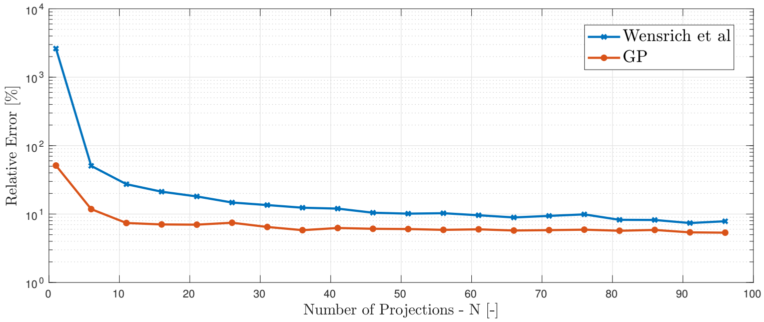


Figure 5: Relative error in the GP-reconstruction as a function of the number of projections.

The result is shown in Fig. 5, which also contains the corresponding curve from [35] for comparison. It is seen that the GP-reconstruction is more accurate and requires less measurements to achieve an equivalent performance.

5.2 Real data

In this section we are solving the problem using data collected from a real-world experiment at the Japan Proton Research Accelerator Complex (JPARC). A brief description of the experimental settings is given below, and we refer to [6] for details.

The sample considered is a thin C-shaped steel plate subject to a compressive load of roughly 7 kN. In polar coordinates it is defined by $r_{\text{in}} = 3.5\,\text{mm} < r < r_{\text{out}} = 10\,\text{mm}$ and $45° < \theta < 315°$.

The data set consists of 86 projections taken from evenly distributed angles around the sample. The experimental resolution of the detector is 512x512. Since this particular experiment was dealing with a planar strain field the detector pixels could be grouped by column (in the out of plane direction) to improve the statistics, giving a possible 512 measurements per projection. However, due to the sample size and shape not all rays would have passed through the sample. On average roughly 350 measurements were made per projection, giving a total amount of nearly 30 000 measurements. It should be noted that the sample took up slightly less than half the detectors height (as another sample was also being analysed) and so about 200 pixels were binned for each measurement.

The Matérn$_{5/2}$ covariance function (34a) was used, and for the approximate settings we have taken $\rho_x = \rho_y = 2.5 r_{\text{out}}$ and a total number of 673 basis functions.

The result is shown in Fig. 6. The top row shows a finite element solution to the simulated problem, while the middle and bottom rows contains the mean and standard deviation of the GP reconstruction. Fig. 7 is showing the ϵ_{yy}- and ϵ_{xx}-components along the line $y = 0$. The data points labelled KOWARI emerges from a constant wavelength scanning experiment, which is a well established method for accurate measurement of average strain within a gauge volume [6]. Also, LS denote the reconstruction obtained from the least squares approach used by [6].

Although the GP reconstruction follows the overall structure of the FEA solution and the KOWARI data, there are regions of notable deviations. This is particularly clear at the right sides in Figure 7, near the boundary of the sample. It has been observed that this deviating behaviour is present when the reconstruction is made from simulated data as well. The boundary challenges are intuitively understood from the nature of the model. From the GP's perspective, the inferred function is a continuous object, and the natural problem boundary of the sample is not built into the model. Outside the sample the reconstruction will fall back to the prior mean, which obviously has a higher impact on the boundary than on the interior. This effect gives rise to a perceived non-smooth feature, which is hard to capture with the relatively smooth covariance function provided by Matérn$_{5/2}$.

Moreover, it is not obvious what settings to choose for the approximation method. In theory, the approximation improves as the number of basis functions is increased. In practice, too many basis functions entails numerical problems, while too few gives a poor approximation. This trade-off requires a somehow ad hoc user selection and the precise impact on the solution is hard to anticipate.

Data-specific error sources related to the collection and processing of the raw data are discussed more thoroughly by [6]. For example, certain ray paths are short relative to others, and the Bragg-edge estimate hence becomes less accurate in these cases. See [32] for more discussions on systematic error sources in this context.

6 Concluding remarks

6.1 Compatibility constraint

If the strain field can be expressed as emerging from a continuous, single-valued displacement field, then it is called *compatible*. Compatibility can be expressed as the linear constraint

$$\underbrace{\begin{bmatrix} \frac{\partial^2}{\partial y^2} & -\frac{\partial^2}{\partial x \partial y} & \frac{\partial^2}{\partial x^2} \end{bmatrix}}_{\mathscr{F}_{\mathbf{x}}^{C}} \mathbf{f} = 0.$$

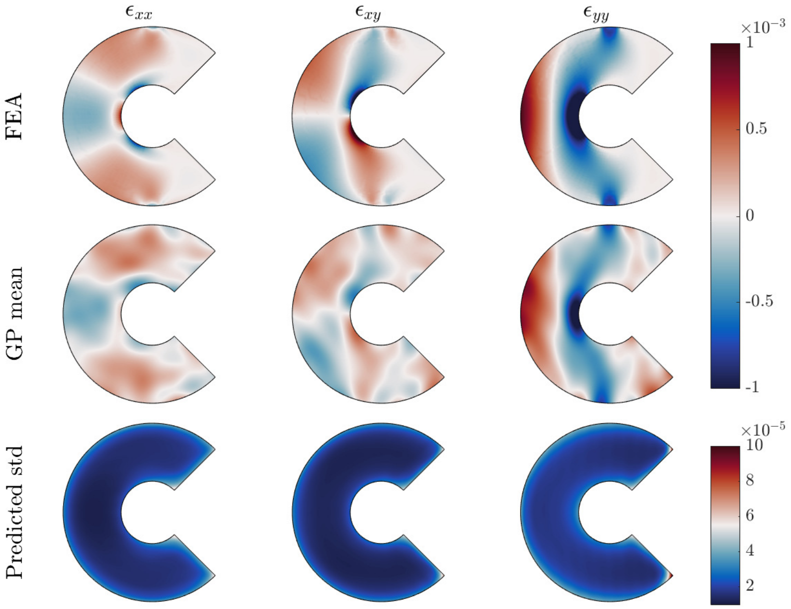


Figure 6: Top row: simulated solution obtained through finite element analysis. Middle and bottom row: mean and standard deviation of the GP reconstruction.

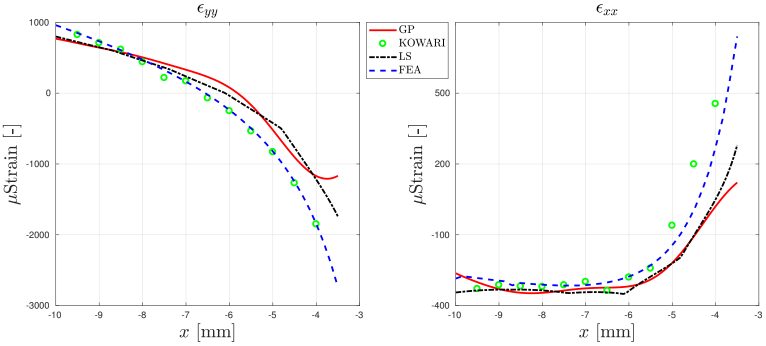


Figure 7: Plots of the GP reconstruction along the line $y = 0$, compared to constant wavelength scanning (KOWARI), a finite element simulation (FEA) and a reconstruction obtained through the least squares approach (LS).

If we would like to construct a model with this constraint imposed as well, the scalar function $\varphi(\mathbf{x})$ can no longer be chosen arbitrary. Instead it will be governed by the relation

$$\mathscr{F}_{\mathbf{x}}^{C}\mathcal{L}_{\mathbf{x}}\varphi = 0, \tag{36}$$

where $\mathcal{L}_{\mathbf{x}}$ is given by (8). This is the strain formulation of the *biharmonic equation*, with solutions called *biharmonic functions*. As shown by [18], separation of variables can be used to express a general solution φ^*, where the specific form is determined by the boundary conditions of the problem, and so is the number of parameters.

This gives rise to at least two questions. Firstly, if there is a parametric form of the solution, why would we use a non-parametric regression method? The answer is that a parametric model require the exact form of the solution so that the number of parameters to be estimated is known. This can only be done provided accurate information of the boundary conditions, which in general can not be assumed to be available. Furthermore, for some boundary conditions the analytical solution may have to be expressed as an infinite sum, thus involving an infinite number of parameters, which makes a parametric model unsuitable.

The second question is, can we include the knowledge of the general solution in the GP model? In theory, the answer is yes. Let $\varphi^* = \varphi^*_{\mathbf{w}}$ denote that the function is parametrised with the parameters in the vector $\mathbf{w}$. Treating these parameters as random variables and assigning them the distribution $p(\mathbf{w})$ allows us to calculate the covariance function of $\varphi^*_{\mathbf{w}}$

$$k_{\varphi^*_{\mathbf{w}}}(\mathbf{x}, \mathbf{x'}) = \int \varphi^*_{\mathbf{w}}(\mathbf{x})\varphi^*_{\mathbf{w}}(\mathbf{x}')p(\mathbf{w})d\mathbf{w}. \tag{37}$$

To be general, we must assume $\mathbf{w}$ to be infinite-dimensional. This fact itself does not imply that an analytical expression could not be found, but the specific form that φ^* requires may not allow it. In any case, we are leaving this as a potential topic of further research.

6.2 Samples with grain texture

The GP prior was designed such that any strain fields generated would automatically satisfy the equilibrium constraints as written in (7). These equilibrium equations are valid for planar, isotropic, linearly elastic samples under the assumption of plane stress. Although the framework presented for strain tomography using GPs does not explicitly prevent the adaptation of this model to allow for samples with strong grain texture, there are significant challenges. These challenges are present both in the acquisition of transmission strain measurements and in the reconstruction from these measurements. During measurement acquisition strong texture may result in the Bragg-edge of interest being unobservable from particular measurement directions, however current research into full pattern fitting may provide a solution to this problem in the future [22–24]. During reconstruction, texture is problematic as different strain measurements may relate to the material bulk strain in different ways. If this relationship can be quantified it may be possible to build it into the GP model.

6.3 Relation to diffraction measurements

Since comparison with a diffraction-based method (KOWARI) was made in Section 5.2, we would like to add a brief comment on the relation between these methods. We would argue that tomographic methods such as the one presented in our paper and diffraction measurements are complementary rather than opposing.

It is important to recognise the differences between these methods. Diffraction-based methods provide measurements of average strain within a gauge volume at defined locations, whereas tomographic methods provide the full strain field over the sample. In addition, future increases in beam power at JPARC and other facilities should reduce the time required to collect Bragg-edge transmission measurements. Under these conditions, one could expect to see a significant reduction in time required to determine the full strain field

Therefore we would suggest that if a particular area of interest is known, diffraction measurements may be a good choice, whereas if it is not known and the user would like to analyse the full field the method presented in this paper may be preferable.

6.4 Future work

First of all, recall that this paper only considers the two-dimensional problem. Since the real world is three-dimensional, it is natural to extend the method accordingly. The main difference is that the target function (6) becomes six-dimensional since a three-dimensional strain field has six unique components. Following this, the equilibrium constraints take an extended form, which obviously is reflected in the covariance model. However, the challenges are most likely not conceptual, but rather limited to the implementation.

Another interesting topic relates to the discussion in Section 6.1, where we introduced the idea of constructing a covariance function for compatible strain fields based on the theoretical solution. Although this appears to be intractable and perhaps not desirable, we did not investigate the subject any further. However, there may be situations in which a theoretical formulation can be derived in an exact or approximate form when this idea may be applicable – especially if the strain field is governed by relatively simple equations.

Furthermore, one may want to explore other covariance models. As always, simple alternatives such as the squared exponential and the Matérn covariance functions should always be tried first, since they have shown to perform well in many applications. As for the particular problems we have considered, there has not been enough indications motivating the implementation of a more advanced option. Also, non-stationary alternatives as discussed below can not be used with the approximation method employed in this paper.

However, it may be situations where extensions are necessary. An example is if the inferred function contains significant non-smooth features, such as rapid changes or discontinuities. In those cases it may be hard to obtain a satisfying reconstruction with a simple model – this is referred to as *misspecification.* As shown by [17], such situations can be dealt with by building the covariance function as a sum of several terms encoding different properties, or changing to another covariance function that is better suited to the data. Even more powerful models having gained interest in recent years are the so-called *deep GPs* [2] and the related *manifold GPs* [1].

Particularly, in some situations the strain behaviour varies significantly between different well-specified parts of the domain, although nearby located. An example of this is the ball bearing problem illustrated by [36]. A way to deal with this might be to use different GPs in each subdomain, conceptually similar to the piecewise GP approach employed by [27] or a so-called *mixtures of experts* model [31]. The most challenging part here is to find a neat way of making use of data spanning multiple subdomains, as is the case with line integral measurements. An extended covariance model as outlined above is a potential alternative for this problem as well. For detailed discussions on the model selection problem, see e.g. [17].

6.5 Conclusion

In this paper we have introduced the concept of probabilistic modelling within the field of tomographic reconstruction. In particular, we have shown that Gaussian processes can be used for strain field estimation from Bragg-edge measurements. The probabilistic nature of the model allows for a systematic treatment of the noise and it provides a direct uncertainty measure of the reconstruction. We have shown that known physical laws can be explicitly incorporated in the design of the associated covariance function, relying on the property that Gaussian processes are closed under linear transformations. Experiments performed on simulated and real data indicates that the method has a high potential which opens up for other tomographic applications as well.

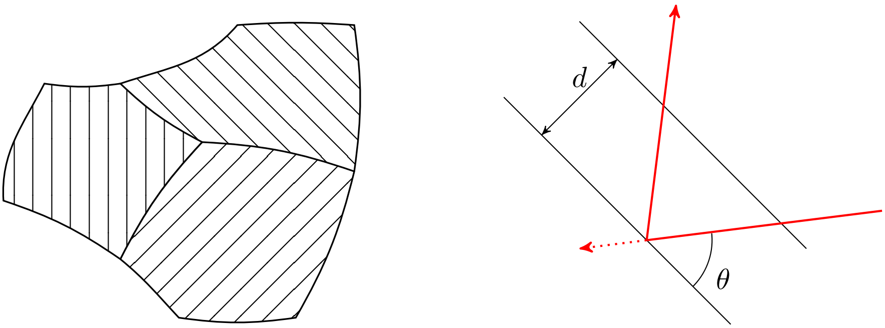


Figure 8: *Left*: Simple illustration of a crystal structure. The material is built up by crystals with planes of certain lattice spacing directed in different angles. *Right*: A ray incident on the material will interact with the crystals whose planes are directed in an angle such that Bragg's law (38) is fulfilled.

7 Acknowledgements

This research was financially supported by the Swedish Foundation for Strategic Research (SSF) via the project *ASSEMBLE* (contract number: RIT15-0012).

A Bragg-edge method

A well-established method for strain estimation within deformed polycrystalline materials relies upon so-called Bragg-edge analysis [20]. A summary of the procedure goes as follows.

The sample investigated is penetrated by neutron beams in a two-step procedure – before and after the deformation occurs. The neutron beams contains a spectra of wavelengths, and they are transmitted in pulses each with a well-known relation between wavelength and intensity. After having passed through the material, the intensity of the beams is recorded at a detector. The wavelength profiles can be measured because of the direct relationship between velocity and wavelength and hence the recorded arrival time of the neutron at the detector is a proxy for the wavelength.

The material contains a very large number of randomly oriented crystal planes that the neutrons interact with, and constructive diffraction occurs according to *Bragg's law*

$$\lambda = 2d \sin\theta, \tag{38}$$

where λ is the neutron wavelength, d is the lattice spacing between the crystal planes and θ is the scattering angle, see Fig. 8. The neutrons are scattered up until $\theta = 90°$, a point at which they are reflected back towards the incoming direction – so-called *backscattering*. For larger wavelengths, no scattering can occur which results in a sudden increase in the relative transmission rate (the ratio of the open beam intensity when no sample is present and the measured intensity when the sample is present). This is known as a *Bragg-edge*. The change in position of the Bragg-edges due to the deformation of the sample is used to calculate a measure of the average strain $\langle\epsilon\rangle$ along the propagating direction of the neutron beam

$$\langle\epsilon\rangle = \frac{d - d_0}{d_0}, \tag{39}$$

where d_0 and d denote the lattice spacings before and after deformation, respectively. Since the material consists of a large number of lattice spacings, each measurement contains several Bragg-edges. In practice a Bragg-edge is chosen that is characteristic of the materials bulk properties (elastic modulus) and also has a good Bragg-edge height (dependent on the source spectra etc). A measurement of the form (39) is modelled with the LRT (2).

B Gaussian processes under linear transformations

A useful property of the GP is that it is closed under linear functional transformations [4, 7, 14, 17, 33]. This means that if

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x'})\big),$$

then

$$L_{\mathbf{w}}[f(\mathbf{x})] \sim \mathcal{GP}\Big(L_{\mathbf{w}}[m(\mathbf{x})], L^2_{\mathbf{w},\mathbf{w'}}[k(\mathbf{x}, \mathbf{x'})]\Big),$$

where $L_{\mathbf{w}}$ is a linear functional with argument $\mathbf{w}$, and $L^2_{\mathbf{w},\mathbf{w'}}$ indicates that it is acting on both arguments of k. With $L_{\mathbf{w}}$ being linear we mean that

$$L_{\mathbf{w}}[\alpha f(\mathbf{x}) + \beta g(\mathbf{x})] = \alpha L_{\mathbf{w}}[f(\mathbf{x})] + \beta L_{\mathbf{w}}[g(\mathbf{x})], \tag{40}$$

for the two scalars α and β. Two common and important examples of linear functionals are differentiation

$$\mathcal{D}_{i,\boldsymbol{\zeta}}[f] = \frac{\partial f}{\partial x_i}\Big|_{\mathbf{x}=\boldsymbol{\zeta}}, \tag{41}$$

and integration

$$\mathcal{I}_{\Omega}[f] = \int_{\Omega} f(\mathbf{x}) d\mathbf{x}. \tag{42}$$

The key here, which makes the closure property so useful, is that the function and the functional have a joint Gaussian distribution. This implies that predictions of the function can be conditioned on observations of the functional, and vice versa. This property is useful in regression problems where we can not observe the function directly.

For example, consider integration of a one-dimensional function $f(x)$ over the interval $\Omega = [a\ b]$. If we model the function with a GP

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \tag{43}$$

then it follows from the above that

$$\begin{aligned} L_{\mathbf{w}=\Omega}[f] &= \mathcal{I}_\Omega[f] = z(\Omega) \\ &= \int_a^b f(x)dx \sim \mathcal{GP}\left(\int_a^b \mu(x)dx, \int_{a'}^{b'}\int_a^b k(x, x')dxdx'\right). \end{aligned} \tag{44}$$

Note that the input to this GP is not the variable x, but the parameterisation of the integration interval Ω. More concretely, assume that we want to predict the value of $f(x_*)$ from integral measurements of $f(x)$. An element in the Gram matrix then becomes

$$\mathbf{K}_{ij} = \int_{a_j}^{b_j}\int_{a_i}^{b_i} k(x, x')dxdx', \tag{45}$$

which describes the correlation between $\int_{a_i}^{b_i} f(x)dx$ and $\int_{a_j}^{b_j} f(x')dx'$, respectively. We then build the vector $\mathbf{k}_*$ according to

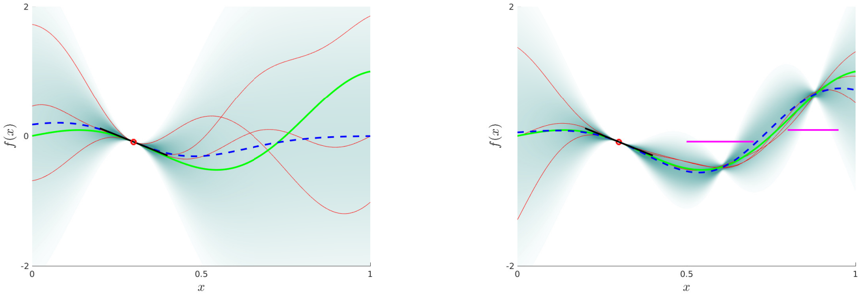
$$(\mathbf{k}_*)_i = \int_{a_i}^{b_i} k(x_*, x')dx', \tag{46}$$

which is the correlation between $\int_{a_i}^{b_i} f(x)dx$ and the function value $f(x_*)$. The GP regression is performed as usual

$$\mathbb{E}[f_*|\mathbf{y}] = \mathbf{k}_*^\mathsf{T}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}, \tag{47a}$$

$$\mathbb{V}[f_*|\mathbf{y}] = k(x_*, x_*) - \mathbf{k}_*^\mathsf{T}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_*. \tag{47b}$$

An example of GP regression using functional observations is shown in Fig. 9, where noise-free observations have been generated from the function $f(x) = x\cos 2\pi x$, shown as the solid thick green line. The observations consists of one function measurement (red circle), one derivative (tangential solid thick black line) and two integrals (horizontal thick pink lines). The squared exponential covariance function $k(x, x') = \sigma_f^2 e^{-l^{-2}(x-x')^2/2}$ is used with $\sigma_f = 1$ and $l = 0.2$. Notice that the mean prediction as well as the samples obey the observed properties (although for the integrals this is not directly seen in the plot).



(a) Posterior after one function observation and one derivative observation.

(b) Posterior after one function observation, one derivative observation and two integral observations.

Figure 9: Example of GP regression with functional observations. The regression is done using one function observation (red circle), one derivative observation (tangential solid thick black line) and two integral observations (horizontal thick pink lines).

The procedure is easily extended to vector valued functions [21]. Letting $\mathcal{L}_{\mathbf{x}}$ denote the transformation (such that an element in $\mathcal{L}_{\mathbf{x}}$ is a linear operator), then we have that

$$\mathcal{L}_{\mathbf{x}}\mathbf{f} \sim \mathcal{GP}(\mathcal{L}_{\mathbf{x}}\mathbf{m}(\mathbf{x}), \mathcal{L}_{\mathbf{x}}\mathbf{K}_{\mathbf{f}}(\mathbf{x}, \mathbf{x}')\mathcal{L}_{\mathbf{x}'}^{\mathsf{T}}). \tag{48}$$

Since the notation might suggest otherwise, we should emphasise that all operators in $\mathcal{L}_{\mathbf{x}}\mathbf{K}(\mathbf{x}, \mathbf{x}')\mathcal{L}_{\mathbf{x}'}^{\mathsf{T}}$ are applied to the elements in $\mathbf{K}$ from the right. As a simple example, assume that the function $\mathbf{f}(\mathbf{x}) : \mathbb{R}^2 \mapsto \mathbb{R}^2$ is modelled with a GP

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{\mathbf{f}}(\mathbf{x}, \mathbf{x}')), \tag{49}$$

where

$$\mathbf{K}_{\mathbf{f}}(\mathbf{x}, \mathbf{x}') = \begin{bmatrix} k_{11}(\mathbf{x}, \mathbf{x}') & k_{12}(\mathbf{x}, \mathbf{x}') \\ k_{21}(\mathbf{x}, \mathbf{x}') & k_{22}(\mathbf{x}, \mathbf{x}') \end{bmatrix}. \tag{50}$$

Assume further that

$$\mathbf{g} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \end{bmatrix} \mathbf{f}. \tag{51}$$

Now it follows from (48) that $\mathbf{g}$ is also a GP

$$\mathbf{g}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{m}_{\mathbf{g}}, \mathbf{K}_{\mathbf{g}}(\mathbf{x}, \mathbf{x}')),$$

where

$$\mathbf{m_g} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \end{bmatrix} \mathbf{0} = \mathbf{0}, \tag{52}$$

and

$$\begin{aligned} \mathbf{K_g}(\mathbf{x}, \mathbf{x}') &= \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} k_{11}(\mathbf{x}, \mathbf{x}') & k_{12}(\mathbf{x}, \mathbf{x}') \\ k_{21}(\mathbf{x}, \mathbf{x}') & k_{22}(\mathbf{x}, \mathbf{x}') \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x'} & 0 \\ 0 & \frac{\partial}{\partial y'} \end{bmatrix}^{\mathsf{T}} \\ &= \begin{bmatrix} \frac{\partial^2}{\partial x^2} k_{11}(\mathbf{x}, \mathbf{x}') & \frac{\partial^2}{\partial x \partial y'} k_{12}(\mathbf{x}, \mathbf{x}') \\ \frac{\partial^2}{\partial y \partial x'} k_{21}(\mathbf{x}, \mathbf{x}') & \frac{\partial^2}{\partial y^2} k_{22}(\mathbf{x}, \mathbf{x}') \end{bmatrix}. \end{aligned} \tag{53}$$

Using a covariance function with this particular structure ensures that the relation (51) is fulfilled.

C Building the covariance function using the Airy stress function

To ensure a physical solution, our model should obey the essential *equilibrium constraints*, which must be fulfilled by the strain field at all points. As described by [18, p. 132], the equilibrium constraints for a two dimensional *stress* field are given by

$$\frac{\partial \tilde{\sigma}_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} = 0, \tag{54a}$$

$$\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \tilde{\sigma}_{yy}}{\partial y} = 0, \tag{54b}$$

where $\tilde{\sigma}_{xx} = \sigma_{xx} - V$, $\tilde{\sigma}_{yy} = \sigma_{yy} - V$ and V is a potential function. These equations are satisfied by letting the components be represented as

$$\tilde{\sigma}_{xx} = \frac{\partial^2 \varphi}{\partial y^2}, \tag{55a}$$

$$\sigma_{xy} = -\frac{\partial^2 \varphi}{\partial x \partial y}, \tag{55b}$$

$$\tilde{\sigma}_{yy} = \frac{\partial^2 \varphi}{\partial x^2}, \tag{55c}$$

where the arbitrary scalar function $\varphi = \varphi(x, y)$ is the so-called *Airy stress function.* Letting $V = 0$ to keep the notation uncluttered, we have that

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{yy} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2}{\partial y^2} \\ -\frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial x^2} \end{bmatrix} \varphi. \tag{56}$$

Applying Hooke's law for isotropic linearly elastic solid materials

$$\begin{bmatrix}\epsilon_{xx}\\ \epsilon_{xy}\\ \epsilon_{yy}\end{bmatrix} = \frac{1}{E}\begin{bmatrix}1 & 0 & -\nu\\ 0 & 1+\nu & 0\\ -\nu & 0 & 1\end{bmatrix}\begin{bmatrix}\sigma_{xx}\\ \sigma_{xy}\\ \sigma_{yy}\end{bmatrix}, \tag{57}$$

where E and ν denote Young's modulus and Poisson's ratio, respectively, we end up with

$$\begin{aligned}\mathbf{f} = \begin{bmatrix}\epsilon_{xx}\\ \epsilon_{xy}\\ \epsilon_{yy}\end{bmatrix} &= \frac{1}{E}\begin{bmatrix}1 & 0 & -\nu\\ 0 & 1+\nu & 0\\ -\nu & 0 & 1\end{bmatrix}\begin{bmatrix}\frac{\partial^2}{\partial y^2}\\ -\frac{\partial^2}{\partial x \partial y}\\ \frac{\partial^2}{\partial x^2}\end{bmatrix}\varphi\\ &= \frac{1}{E}\begin{bmatrix}\frac{\partial^2}{\partial y^2} - \nu\frac{\partial^2}{\partial x^2}\\ -(1+\nu)\frac{\partial^2}{\partial x \partial y}\\ \frac{\partial^2}{\partial x^2} - \nu\frac{\partial^2}{\partial y^2}\end{bmatrix}\varphi = \mathcal{L}_{\mathbf{x}}\varphi. \end{aligned}\tag{58}$$

Let us now model φ as a GP

$$\varphi \sim \mathcal{GP}(0, k_\varphi(\mathbf{x}, \mathbf{x}')). \tag{59}$$

Since the strain field $\mathbf{f}$ is mapped from φ through the linear functional $\mathcal{L}_{\mathbf{x}}$, it follows from Section B that it is also a GP

$$\mathbf{f} \sim \mathcal{GP}(\mathbf{0}, \mathcal{L}_{\mathbf{x}}k_\varphi(\mathbf{x}, \mathbf{x}')\mathcal{L}_{\mathbf{x}'}^{\mathsf{T}}). \tag{60}$$

Building the covariance function this way will by construction guarantee that any sample drawn from the resulting posterior fulfils the equilibrium constraints given by (54).

D Details on approximative computations

D.1 Elements of the Q-matrix

As stated in (8), the target function $\mathbf{f}(\mathbf{x})$ is related to the scalar function $\varphi(x, y)$ according to

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix}\frac{\partial^2}{\partial y^2} - \nu\frac{\partial^2}{\partial x^2}\\ -(1+\nu)\frac{\partial^2}{\partial x \partial y}\\ \frac{\partial^2}{\partial x^2} - \nu\frac{\partial^2}{\partial y^2}\end{bmatrix}\varphi(x, y) = \mathcal{L}_{\mathbf{x}}\varphi(x, y). \tag{61}$$

In the approximative method described in Section 2.6, we are projecting the functional (16) onto the basis functions

$$\phi(\mathbf{x}) = \frac{1}{\sqrt{\rho_x \rho_y}}\sin\big(\lambda_x(x+\rho_x)\big)\sin\big(\lambda_y(y+\rho_y)\big). \tag{62}$$

To keep the notation uncluttered, we are in this section omitting the indexing of the basis functions as well as the measurements, but keep in mind that each calculation described by the equations below must be repeated m times for each measurement.

Each measurement with inputs $\mathbf{x}^0$, L and $\hat{\mathbf{n}}$ requires us to calculate the integral

$$\frac{1}{L}\int_0^L \arg(x^0 + sn_x, y^0 + sn_y)ds, \tag{63}$$

where

$$\arg(x,y) = \vec{\mathbf{n}}^{\mathsf{T}}\mathcal{L}_{\mathbf{x}}\phi(x,y) = \\ \left[n_x^2\left(\frac{\partial^2}{\partial y^2} - \nu\frac{\partial^2}{\partial x^2}\right) - 2n_xn_y(1+\nu)\frac{\partial^2}{\partial x\partial y} + n_y^2\left(\frac{\partial^2}{\partial x^2} - \nu\frac{\partial^2}{\partial y^2}\right)\right]\phi(x,y). \tag{64}$$

The partial derivatives involved have the following explicit forms

$$\frac{\partial^2}{\partial x^2}\phi(x,y) = -\lambda_x^2\phi(x,y), \tag{65a}$$

$$\frac{\partial^2}{\partial x\partial y}\phi(x,y) = \lambda_x\lambda_y\underbrace{\frac{1}{\sqrt{\rho_x\rho_y}}\cos\big(\lambda_x(x+\rho_x)\big)\cos\big(\lambda_y(y+\rho_y)\big)}_{\phi_c(x,y)}, \tag{65b}$$

$$\frac{\partial^2}{\partial y^2}\phi(x,y) = -\lambda_y^2\phi(x,y). \tag{65c}$$

Substituting the expressions (65) into (64) yields

$$\arg(x,y) = \underbrace{\left[n_x^2\left(\nu\lambda_x^2 - \lambda_y^2\right) + n_y^2\left(\nu\lambda_y^2 - \lambda_x^2\right)\right]}_{C_S}\phi(x,y) \\ - \underbrace{2(1+\nu)n_xn_y\lambda_x\lambda_y}_{C_O}\phi_C(x,y). \tag{66}$$

We can now see that (63) involves calculation of the two integrals

$$I_1 = \int_0^L \phi(x+sn_x, y+sn_y)ds \\ = \frac{1}{2\sqrt{\rho_x\rho_y}}\left[\frac{1}{\Lambda^-}\sin(\Lambda^- s + B^-) - \frac{1}{\Lambda^+}\sin(\Lambda^+ s + B^+)\right]_{s=0}^{s=L}, \tag{67a}$$

$$
\begin{aligned}
I_2 &= \int_0^L \phi_C(x + sn_x, y + sn_y) ds \\
&= \frac{1}{2\sqrt{\rho_x \rho_y}} \left[\frac{1}{\Lambda^-} \sin(\Lambda^- s + B^-) + \frac{1}{\Lambda^+} \sin(\Lambda^+ s + B^+) \right]_{s=0}^{s=L}, \quad (67b)
\end{aligned}
$$

where we have defined

$$
\Lambda^{\pm} = n_x \lambda_x \pm n_y \lambda_y, \quad (68a)
$$

$$
B^{\pm} = \lambda_x(x^0 + \rho_x) \pm \lambda_y(y^0 + \rho_y). \quad (68b)
$$

Finally, we end up with

$$
\frac{1}{L} \int_0^L \arg(x^0 + sn_x, y^0 + sn_y) ds = \frac{C_S I_1 - C_O I_2}{L}. \quad (69)
$$

Hence, the element $\mathbf{Q}_{ij}$ in (29) is obtained by in the above calculations use the i:th basis function with corresponding eigenvalues, and the input arguments of the j:th measurement.

D.2 Marginal Likelihood Expressions

By replacing $\mathbf{K}_\text{I}$ in (31) with the approximation $\mathbf{K}_\text{I} \approx \mathbf{Q}^\mathsf{T} \mathbf{\Lambda} \mathbf{Q}$, and letting $\hat{\mathbf{Q}} = \mathbf{Q}^\mathsf{T} \mathbf{\Lambda} \mathbf{Q} + \sigma^2 I$, we get the following expression for the logarithm of the approximate marginal likelihood

$$
\begin{aligned}
\log p(\mathbf{y} | \{\boldsymbol{\eta}_i\}, \boldsymbol{\theta}) &\approx \log \tilde{p}(\mathbf{y} | \{\boldsymbol{\eta}_i\}, \boldsymbol{\theta}) \\
&= -\frac{1}{2} \log\det(\hat{\mathbf{Q}}) - \frac{1}{2} \mathbf{y}^\mathsf{T} \hat{\mathbf{Q}}^{-1} \mathbf{y} - \frac{N}{2} \log 2\pi. \quad (70)
\end{aligned}
$$

For simplicity, we separate the partial derivative with respect to the noise σ and the partial derivatives with respect to the other hyperparameters:

$$
\frac{\partial}{\partial \theta_i} \log \tilde{p}(\mathbf{y} | \{\boldsymbol{\eta}_i\}, \boldsymbol{\theta}) = -\frac{1}{2} \frac{\partial \log\det(\hat{\mathbf{Q}})}{\partial \theta_i} - \frac{1}{2} \frac{\partial \mathbf{y}^\mathsf{T} \hat{\mathbf{Q}}^{-1} \mathbf{y}}{\partial \theta_i}, \quad (71a)
$$

$$
\frac{\partial}{\partial \sigma} \log \tilde{p}(\mathbf{y} | \{\boldsymbol{\eta}_i\}, \boldsymbol{\theta}) = -\frac{1}{2} \frac{\partial \log\det(\hat{\mathbf{Q}})}{\partial \sigma} - \frac{1}{2} \frac{\partial \mathbf{y}^\mathsf{T} \hat{\mathbf{Q}}^{-1} \mathbf{y}}{\partial \sigma}. \quad (71b)
$$

Introducing $\mathbf{Z} = \sigma^2\mathbf{\Lambda}^{-1} + \mathbf{Q}\mathbf{Q}^\mathsf{T}$, the explicit expressions are:

$$\log\det(\hat{\mathbf{Q}}) = -(N-m)\log\sigma^2 - \log\det(\mathbf{Z}) - \sum_j \log \mathbf{\Lambda}_{jj}, \tag{72a}$$

$$\frac{\partial \log\det(\hat{\mathbf{Q}})}{\partial\theta_i} = -\sum_j \mathbf{\Lambda}_{jj}\frac{\partial \mathbf{\Lambda}_{jj}}{\partial\theta_i} + \sigma^2 tr\mathbf{Z}^{-1}\mathbf{\Lambda}^{-2}\frac{\partial\mathbf{\Lambda}}{\partial\theta_i}, \tag{72b}$$

$$\frac{\partial \log\det(\hat{\mathbf{Q}})}{\partial\sigma} = -2\frac{N-m}{\sigma} - 2\sigma tr\mathbf{Z}^{-1}\mathbf{\Lambda}^{-1}, \tag{72c}$$

$$\mathbf{y}^\mathsf{T}\hat{\mathbf{Q}}^{-1}\mathbf{y} = \frac{1}{\sigma^2}\left(\mathbf{y}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Z}^{-1}\mathbf{Q}\mathbf{y} - \mathbf{y}^\mathsf{T}\mathbf{y}\right), \tag{72d}$$

$$\frac{\partial \mathbf{y}^\mathsf{T}\hat{\mathbf{Q}}^{-1}\mathbf{y}}{\partial\theta_i} = \mathbf{y}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Z}^{-1}\left[\mathbf{\Lambda}^{-2}\frac{\partial\mathbf{\Lambda}}{\partial\theta_i}\right]\mathbf{Z}^{-1}\mathbf{Q}\mathbf{y}, \tag{72e}$$

$$\frac{\partial \mathbf{y}^\mathsf{T}\hat{\mathbf{Q}}^{-1}\mathbf{y}}{\partial\sigma} = -\frac{2}{\sigma}\mathbf{y}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Z}^{-1}\mathbf{\Lambda}^{-1}\mathbf{Z}^{-1}\mathbf{Q}\mathbf{y} + \frac{2}{\sigma^3}\mathbf{y}^\mathsf{T}\mathbf{y} - \frac{2}{\sigma^3}\mathbf{y}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Z}^{-1}\mathbf{Q}\mathbf{y}. \tag{72f}$$

References

[1] R. Calandra, J. Peters, C. E. Rasmussen and M. P. Deisenroth. "Manifold Gaussian Processes for Regression". In: *In Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. 2016, pp. 3338–3345.

[2] A. C. Damianou. "Deep Gaussian processes and variational propagation of uncertainty". PhD thesis. Department of neuroscience, University of Sheffield, 2015.

[3] M. Fitzpatrick and A. Lodini. *Analysis of Residual Stress by Diffraction Using Neutron and Synchrotron Radiation*. CRC Press, Jan. 2003.

[4] R. Garnett. *Lecture 11: Bayesian Quadrature*. University Lecture. Accessed: 2019-05-01. Available from: `http://www.cse.wustl.edu/~garnett/cse515t/spring_2018/files/lecture_notes/11.pdf`. 2018.

[5] Z. Ghahramani. "Probabilistic machine learning and artificial intelligence". In: *Nature* 521 (2015), pp. 452–459.

[6] J. N. Hendriks, A. W. T. Gregg, C. M. Wensrich, A. S. Tremsin, T. Shinohara, M. Meylan, E. H. Kisi, V. Luzin and O. Kirsten. "Bragg-edge elastic strain tomography for in situ systems from energy-resolved neutron transmission imaging". In: *Physical Review Materials* 1.5 (Oct. 2017), p. 053802.

[7] P. Hennig and M. Kiefel. “Quasi-Newton methods: A new direction”. In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 843–865.

[8] C. Jidling, N. Wahlström, A. Wills and T. B. Schön. “Linearly constrained Gaussian processes”. In: *Advances in Neural Information Processing Systems (NIPS)*. Los Angeles, US, 2017, pp. 1215–1224.

[9] A. M. Korsunsky, W. J. Vorster, S. Y. Zhang, D. Dini, D. Latham, M. Golshan, J. Liu, Y. Kyriakoglou and M. J. Walsh. “The principle of strain reconstruction tomography: Determination of quench strain distribution from diffraction measurements”. In: *Acta Materiala* 54.8 (2008), pp. 2101–2108.

[10] A. M. Korsunsky, N. Baimpas, X. Song, J. Belnoue, F. Hofmann, B. Abbey, M. Xie, J. Andrieux, T. Buslaps and T. K. Neo. “Strain tomography of polycrystalline zirconia dental prostheses by synchrotron X-ray diffraction”. In: *Acta Materialia* 59.6 (2011), pp. 2501–2513.

[11] W. R. B. Lionheart and P. J. Withers. “Diffraction tomography of strain”. In: *Inverse Problems* 31.4 (2015), p. 045005.

[12] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2000. ISBN: 9780387987934.

[13] I. Noyan and J. Cohen. *Residual stress: measurement by diffraction and interpretation*. Springer, 2013.

[14] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes*. New York: McGraw-Hill Education, 1991.

[15] M. B. Prime. “Cross-sectional mapping of residual stresses by measuring the surface contour after a cut”. In: *Transactions-American society of mechanical engineers journal of engineering materials and technology* 123.2 (2001), pp. 162–168.

[16] J. Quiñonero Candela, C. E. Ramussen and C. K. I. Williams. “Approximation Methods for Gaussian Process Regression”. In: *Large-scale kernel machines* (Sept. 2007), pp. 203–223.

[17] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT press, Cambridge, MA, 2006.

[18] M. H. Sadd. *Elasticity - Theory, applications and numerics*. New York: Elsevier Butterworth-Heinemann, 2005.

[19] J. R. Santisteban, L. Edwards, M. E. Fitzpatrick, A. Steuwer, P. J. Withers, M. Daymond, M. W. Johnson, N. Rhodes and E. M. Schooneveld. “Strain imaging by Bragg edge neutron transmission”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 481.1 (2002), pp. 765–768.

[20] J. R. Santisteban, L. Edwards, M. E. Fizpatrick, A. Steuwer and P. J. Withers. “Engineering applications of Bragg-edge neutron transmission”. In: *Applied Physics* 74.1 (2002), pp. 1433–1436.

[21] S. Särkkä. “Linear Operators and Stochastic Partial Differential Equations in Gaussian Process Regression”. In: *Proceedings of the Artificial Neural Networks and Machine Learning – ICANN 2011.* Springer, 2011, pp. 151–158.

[22] H. Sato, T. Shinohara, R. Kiyanagi, K. Aizawa, M. Ooi, M. Harada, K. Oikawa, F. Maekawa, K. Iwase, T. Kamiyama et al. “Upgrade of Bragg edge analysis techniques of the RITS code for crystalline structural information imaging”. In: *Physics Procedia* 43 (2013), pp. 186–195.

[23] H. Sato, K. Watanabe, K. Kiyokawa, R. Kiyanagi, K. Hara, T. Kamiyama, M. Furusaka, T. Shinohara and Y. Kiyanagi. “Further improvement of the RITS code for pulsed neutron Bragg-edge transmission imaging”. In: *Physics Procedia* 88 (2017), pp. 322–330.

[24] H. Sato, T. Kamiyama and Y. Kiyanagi. “A Rietveld-type analysis code for pulsed neutron Bragg-edge transmission imaging and quantitative evaluation of texture and microstructure of a welded α-iron plate”. In: *Materials transactions* 52.6 (2011), pp. 1294–1302.

[25] A. Solin and S. Särkkä. “Hilbert space methods for reduced-rank Gaussian process regression”. In: *Statistics and Computing* (Aug. 2019).

[26] Standard, ASTM and others. *Standard test method for determining residual stresses by the hole-drilling strain-gage method.* 2002.

[27] A. Svensson and T. B. Schön. “A flexible state–space model for learning nonlinear dynamical systems”. In: *Automatica* 80 (2017), pp. 189–199.

[28] A. Tremsin, Y. Gao, L. C. Dial, F. Grazzi and T. Shinohara. “Investigation of microstructure in additive manufactured Inconel 625 by spatially resolved neutron transmission spectroscopy”. In: *Science and Technology of Advanced Materials* 17 (July 2016), pp. 324–336.

[29] A. Tremsin, J. McPhate, W. Kockelmann, J. Vallerga, O. H. W. Siegmund and W. B. Feller. "High resolution Bragg edge transmission spectroscopy at pulsed neutron sources: Proof of principle experiments with a neutron counting MCP detector". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors, and Associated Equipment* 633 (May 2011), S235–S238.

[30] A. Tremsin, J. McPhate, A. Steuwer, W. Kockelmann, A. Paradowska, J. Kelleher, J. Vallerga, O. H. W. Siegmund and W. B. Feller. "High-Resolution Strain Mapping Through Time-of-Flight Neutron Transmission Diffraction with a Microchannel Plate Neutron Counting Detector". In: *Strain* 48 (Aug. 2012).

[31] V. Tresp. "Mixtures of Gaussian Processes". In: *Advances in Neural Information Processing Systems (NIPS).* 2001, pp. 654–660.

[32] S. Vogel. "A Rietveld-approach for the analysis of neutron time-of-flight transmission data". PhD thesis. Christian-Albrechts Universitat, Kiel, 2000.

[33] N. Wahlström. "Modeling of Magnetic Fields and Extended Objects for Localization Applications". PhD thesis. Division of Automatic Control, Linköping University, 2015.

[34] T. Watkins, H. Bilheux, K. An, C. A Brice, E. Payzant, R. Dehoff, C. Duty, C. Blue and W. H Peter. "Neutron Characterization for Additive Manufacturing". In: *Advanced Materials and Processes* 171.23 (Mar. 2013).

[35] C. M. Wensrich, J. N. Hendriks, A. Gregg, M. H. Meylan, V. Luzin and A. S. Tremsin. "Bragg-edge neutron transmission strain tomography for in situ loadings". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 383 (2016), pp. 52–58.

[36] C. M. Wensrich, J. N. Hendriks and M. H. Meylan. "Bragg Edge Neutron Transmission Strain Tomography in Granular Systems". In: *Strain* 52.1 (2016), pp. 80–87.

[37] R. Woracek, J. Santisteban, A. Fedrigo and M. Strobl. "Diffraction in neutron imaging—A review". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 878 (2018). Radiation Imaging Techniques and Applications, pp. 141–158.

Paper III

Title

Probabilistic approach to limited-data computed tomography reconstruction.

Authors

Zenith Purisha, Carl Jidling, Niklas Wahlström, Thomas B. Schön, and Simo Särkkä.

Statement of contribution

The method proposed in this paper was developed by the authors jointly. The main part of the work was done by Zenith Purisha, including the production and compilation of experimental results, as well as the majority of the writing. Carl Jidling contributed to the implementation of the GP routines, assisted with parts of the writing and provided feedback during the revision process.

Edited version of

Z. Purisha, C. Jidling, N. Wahlström, T. Schön and S. Särkkä. “Probabilistic approach to limited-data computed tomography reconstruction”. In: *Inverse Problems* 35.10 (2019), p. 105004.

Probabilistic approach to limited-data computed tomography reconstruction

Abstract

In this work, we consider the inverse problem of reconstructing the internal structure of an object from limited x-ray projections. We use a Gaussian process prior to model the target function and estimate its (hyper)parameters from measured data. In contrast to other established methods, this comes with the advantage of not requiring any manual parameter tuning, which usually arises in classical regularisation strategies. Our method uses a basis function expansion technique for the Gaussian process which significantly reduces the computational complexity and avoids the need for numerical integration. The approach also allows for reformulation of come classical regularisation methods as Laplacian and Tikhonov regularisation as Gaussian process regression, and hence provides an efficient algorithm and principled means for their parameter tuning. Results from simulated and real data indicate that this approach is less sensitive to streak artifacts as compared to the commonly used method of filtered backprojection.

1 Introduction

X-ray computed tomography (CT) imaging is a non-invasive method to recover the internal structure of an object by collecting projection data from multiple angles. The projection data is recorded by a detector array and it represents the attenuation of the x-rays which are transmitted through the object. Since the 1960s, CT has been used to a deluge of applications in medicine [9, 10, 19, 26, 30, 41] and industry [2, 8, 11].

Currently, the so-called filtered back projection (FBP) is the reconstruction algorithm of choice because it is very fast [7, 22]. This method requires dense sampling of the projection data to obtain a satisfying image reconstruction. However, for some decades, the limited-data x-ray tomography problem has been a major concern in, for instance, the medical imaging community. The limited data case—also referred to as *sparse projections*—calls for a good solution for several important reasons, including:

- the needs to examine a patient by using low radiation doses to reduce the risk of malignancy or to *in vivo* samples to avoid the modification of the properties of living tissues,

- geometric restrictions in the measurement setting make it difficult to acquire the complete data [36], such as in *mammography* [32, 34, 48, 49] and electron imaging [12],

- the high demand to obtain the data using short acquisition times and to avoid massive memory storage, and

- the needs to avoid—or at least minimise the impact of—the moving artefacts during the acquisition.

Classical algorithms—such as FBP—fail to generate good image reconstruction when dense sampling is not possible and we only have access to limited data. The under-sampling of the projection data makes the image reconstruction (in classical terms) an *ill-posed* problem [31]. In other words, the inverse problem is sensitive to measurement noise and modelling errors. Hence, alternative and more powerful methods are required. Statistical estimation methods play an important role in handling the ill-posedness of the problem by restating the inverse problem as a *well-posed extension* in a larger space of probability distributions [21]. Over the years there have been a lot of work on tomographic reconstruction from limited data using statistical methods (see, e.g., [4, 16, 25, 34, 39, 43]).

In the statistical approach, incorporation of *a priori* knowledge is a crucial part in improving the quality of the image reconstructed from limited projection data. That can be viewed as an equivalent of the regularisation parameter in classical regularisation methods. However, statistical methods, unlike classical regularisation methods, also provide a principled means to estimate the parameters of the prior (i.e., the hyperparameters) which corresponds to automatic tuning of regularisation parameters.

In our work we build the statistical model by using a Gaussian process model [35] with a hierarchical prior in which the (hyper)parameters in the prior become part of the inference problem. As this kind of hierarchical prior can be seen as an instance of a Gaussian process (GP) regression model, the computational methods developed for GP regression in machine learning context [35] become applicable. It is worth noting that some works on employing GP methods for tomographic problems have also appeared before. An iterative algorithm to compute a maximum likelihood point in which the prior information is represented by GP is introduced in [46]. In [18, 20], tomographic reconstruction using GPs to model the strain field from neutron Bragg-edge measurements has been studied. Tomographic inversion using GP for plasma fusion and soft x-ray tomography have been done in [27, 45]. Nevertheless, the proposed approach is different from the existing work.

Our aim is to employ a hierarchical Gaussian process regression model to reconstruct the x-ray tomographic image from limited projection data. Due to the measurement model involving line integral computations, the direct GP approach does not allow for closed form expressions. The first contribution of this article is to overcome this issue by employing the basis function expansion method proposed in [44], which makes the line integral computations tractable as it detaches the integrals from the model parameters. This approach can be directly used for common GP regression covariance functions such as Matérn or squared exponential. The second contribution of this article is to point out that the we can also reformulate classical regularisation, in particular Laplacian and Tikhonov regularization, as Gaussian process regression where only the spectral density of the process (although not the covariance function itself) is well defined. As the basis function expansion only requires the availability of the spectral density, we can build a hierarchical model off a classical regularisation model as well and have a principles means to tune the regularisation parameters. Finally, the third contribution is to present methods for hyperparameter estimation that arise from the machine learning literature and apply the methodology to the tomographic reconstruction problem. In particular, the proposed methods are applied to simulated 2D chest phantom data available in MATLAB and real carved cheese data measured with μCT system. The results show that the reconstruction images created using the proposed GP method outperforms the FBP reconstructions in terms of image quality measured as relative error and as peak signal to noise ratio.

2 Constructing the model

2.1 The tomographic measurement data

Consider a physical domain $\Omega \subset \mathbb{R}^2$ and an attenuation function $f : \Omega \to \mathbb{R}$. The x-rays travel through Ω along straight lines and we assume that the initial intensity (photons) of the x-ray is I_0 and the exiting x-ray intensity is I_d. If we denote a ray through the object as function $s \mapsto (x_1(s), x_2(s))$ Then the formula for the intensity loss of the x-ray within a small distance ds is given as:

$$\frac{dI(s)}{I(s)} = -f(x_1(s), x_2(s))ds, \tag{1}$$

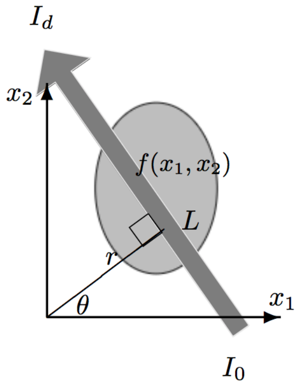


Figure 1: An illustration of the Radon transform. It maps the object f on the (x_1, x_2)-domain into f on the (r, θ) domain. The measurement data is collected from the intensities I_d of x-rays for all lines L through the object $f(x_1, x_2)$ and from different angles of view.

and by integrating both sides of (1), the following relationship is obtained

$$\int_{-R}^{R} f(x_1(s), x_2(s))ds = \log \frac{I_0}{I_d}, \tag{2}$$

where R is the radius of the object or area being examined.

In x-ray tomographic imaging, the aim is to reconstruct f using measurement data collected from the intensities I_d of x-rays for all lines through the object taken from different angles of view. The problem can be expressed using the Radon transform, which can be expressed as

$$\mathcal{R}f(r, \theta) = \int f(x_1, x_2)d\mathbf{x}_L, \tag{3}$$

where $d\mathbf{x}_L$ denotes the 1-dimensional Lebesgue measure along the line defined by $L = \{(x_1, x_2) \in \mathbb{R}^2 : x_1 \cos\theta + x_2 \sin\theta = r\}$, where $\theta \in [0, \pi)$ is the angle and $r \in \mathbb{R}$ is the distance of L from the origin as shown in Figure 1.

The parametrisation of the straight line L with respect to the arc length s can be written as:

$$\begin{aligned} x_1(s, \theta, r) &= r\ \cos(\theta) - s\ \sin(\theta), \\ x_2(s, \theta, r) &= r\ \sin(\theta) + s\ \cos(\theta). \end{aligned} \tag{4}$$

In this work, the object is placed inside a circular disk with radius R. Then, as a function of r and θ the line integral in (3) can be written as

$$\begin{aligned}\mathcal{R}f(r,\theta) &= \int_{-R}^{R} f(x_1(s,\theta,r), x_2(s,\theta,r))\, ds \\ &= \int_{-R}^{R} f(\mathbf{x}^0 + s\hat{\mathbf{u}}) ds,\end{aligned} \tag{5}$$

where

$$\mathbf{x}^0 = \begin{bmatrix} r\cos(\theta) & r\sin(\theta)\end{bmatrix}^{\mathsf{T}}, \qquad \hat{\mathbf{u}} = \begin{bmatrix} -\sin(\theta) & \cos(\theta)\end{bmatrix}^{\mathsf{T}}.$$

In a real x-ray tomography application, the measurement is corrupted by at least two noise types: photons statistics and electronic noise. In x-ray imaging, a massive number of photons are usually recorded at each detector pixel. In such case, a Gaussian approximation for the attenuation data in (2) can be used [3, 37]. Recall that a logarithm of the intensity is involved in (5), and so additive noise is a reasonable model for the electronic noise.

We collect a set of measurements as

$$y_i = \int_{-R}^{R} f(\mathbf{x}_i^0 + s\hat{\mathbf{u}}_i) ds + \varepsilon_i, \tag{6}$$

where i corresponds to the data point index. The corresponding inverse problem is given the noisy measurement data $\{y_i\}_{i=1}^n$ in (6) to reconstruct the object f.

2.2 Gaussian processes as functional priors

A Gaussian process (GP) [35] can be viewed as a distribution over functions, where the function value in each point is treated as a Gaussian random variable. To denote that the function f is modelled as a GP, we formally write

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}),\ k(\mathbf{x},\mathbf{x}')\right). \tag{7}$$

The GP is uniquely specified by the *mean function* $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and the *covariance function* $k(\mathbf{x},\mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$. The mean function encodes our prior belief of the value of f in any point. In lack of better knowledge it is common to pick $m(\mathbf{x}) = 0$, a choice that we will stick to also in this paper.

The covariance function on the other hand describes the covariance between two different function values $f(\mathbf{x})$ and $f(\mathbf{x}')$. The choice of covariance function is the most important part in the GP model, as it stipulates the properties assigned to f. A few different options are discussed in Section 2.4.

As data is collected our belief about f is updated. The aim of regression is to predict the function value $f(\mathbf{x}_*)$ at an unseen test point $\mathbf{x}_*$ by conditioning on the seen data. Consider direct function measurements on the form

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \tag{8}$$

where ε_i is independent and identically distributed (iid) Gaussian noise with variance σ^2, that is, $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. Let the measurements be stored in the vector $\mathbf{y}$. Then the mean value and the variance of the predictive distribution $p(f(\mathbf{x}_*) \mid \mathbf{y})$ are given by [35]

$$\mathbb{E}[f(\mathbf{x}_*) \mid \mathbf{y}] = \mathbf{k}_*^\mathsf{T}(K + \sigma^2 I)^{-1}\mathbf{y}, \tag{9a}$$

$$\mathbb{V}[f(\mathbf{x}_*) \mid \mathbf{y}] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\mathsf{T}(K + \sigma^2 I)^{-1}\mathbf{k}_*. \tag{9b}$$

Here the vector $\mathbf{k}_*$ contains the covariance between $f(\mathbf{x}_*)$ and each measurement while the matrix K contains the covariance between all measurements, such that

$$(\mathbf{k}_*)_i = k(\mathbf{x}_i, \mathbf{x}_*), \tag{10a}$$

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j). \tag{10b}$$

An example of GP regression for a two-dimensional input is given in Figure 2. The red stars indicate the measurements, while the shaded surface is the GP prediction. The blue line highlights a slice of the plot that is shown explicitly to the right, including the 95% credibility region.

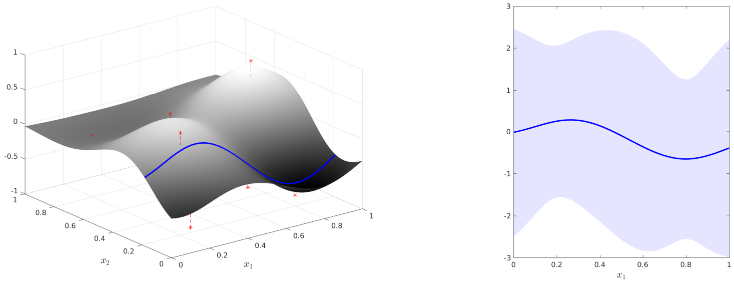


Figure 2: Left: GP prediction (shaded surface) obtained from the measurements (red stars, also indicated by their deviation from the prediction). Right: slice plot of the blue line in the left figure, including the 95% credibility region.

2.3 The Gaussian process for x-ray tomography

In this section, we show how to apply the functional priors presented in Section 2.2 to x-ray tomography application. Since the x-ray measurements

(5) are line integrals of the unknown function $f(\mathbf{x})$, they are linear functionals of the Gaussian process. Hence, we can define a linear functional $\mathcal{H}_{\mathbf{x},i}$ as follows:

$$\mathcal{H}_{\mathbf{x},i} f(\mathbf{x}) = \int_{-R}^{R} f(\mathbf{x}_i^0 + s\hat{\mathbf{u}}_i) ds. \tag{11}$$

and thus the GP regression problem becomes

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}),\ k(\mathbf{x}, \mathbf{x}')\right), \tag{12a}$$

$$y_i = \mathcal{H}_{\mathbf{x},i} f(\mathbf{x}) + \varepsilon_i. \tag{12b}$$

As discussed, for example, in [38, 44] the GP regression equations can be extended to this kind of models, which in this case leads to the following:

$$\mathbb{E}[f(\mathbf{x}_*)|\mathbf{y}] = \mathbf{q}_*^\mathsf{T} (K + \sigma^2 I)^{-1} \mathbf{y}, \tag{13a}$$

$$\mathbb{V}[f(\mathbf{x}_*)|\mathbf{y}] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{q}_*^\mathsf{T} (Q + \sigma^2 I)^{-1} \mathbf{q}_*, \tag{13b}$$

where $\mathbf{y} = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix}^\mathsf{T}$ and

$$(\mathbf{q}_*)_i = \int_{-R}^{R} k(\mathbf{x}_i^0 + s\hat{\mathbf{u}}_i, \mathbf{x}_*) ds, \tag{14a}$$

$$Q_{ij} = \int_{-R}^{R} \int_{-R}^{R} k(\mathbf{x}_i^0 + s\hat{\mathbf{u}}_i, \mathbf{x}_j^0 + s'\hat{\mathbf{u}}_j) ds ds'. \tag{14b}$$

In general we can not expect closed form solutions to (14a)–(14b) and numerical computations are then required. However, even with efficient numerical methods, the process of selecting the hyperparameters is tedious since the hyperparameters are in general not decoupled from the integrand and the integrals need to be computed repeatedly in several iterations. In this paper, we avoid this by using the basis function expansion that will be described in Section 2.6.

2.4 Squared exponential and Matérn covariance functions

An important modelling parameter in Gaussian process regression is the covariance function $k(\mathbf{x}, \mathbf{x}')$ which can be selected in various ways. Because the basis function expansion described in Section 2.6 requires the covariance function to be *stationary*, we here limit our discussion to covariance functions of this form. *Stationarity* means that $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{r})$ where $\mathbf{r} = \mathbf{x} - \mathbf{x}'$, so the covariance only depends on the distance between the input points. In that case we can also work with the spectral density, which is the Fourier transform of the stationary covariance function

$$S(\boldsymbol{\omega}) = \mathcal{F}[k] = \int k(\mathbf{r}) e^{-\mathrm{i}\boldsymbol{\omega}^\mathsf{T}\mathbf{r}} d\mathbf{r}, \tag{15}$$

where again $\mathbf{r} = \mathbf{x} - \mathbf{x}'$.

The perhaps most commonly used covariance function within the machine learning context [35] is the *squared exponential* (SE) covariance function

$$k_{\mathrm{SE}}(\mathbf{r}) = \sigma_f^2 \exp\left[-\frac{1}{2l^2}\|\mathbf{r}\|_2^2\right], \tag{16}$$

which has the following spectral density

$$S_{\mathrm{SE}}(\boldsymbol{\omega}) = \sigma_f^2 (2\pi)^{d/2} l^d \exp\left[-\frac{l^2\|\boldsymbol{\omega}\|_2^2}{2}\right], \tag{17}$$

where d is the dimensionality of $\mathbf{x}$ (in our case $d = 2$). The SE covariance function is characterised by the magnitude parameter σ_f and the *length scale* l. The squared exponential covariance function is popular due to its simplicity and ease of implementation. It corresponds to a process whose sample paths are infinitely many times differentiable and thus the functions modelled by it are very smooth.

Another common family of covariance functions is given by the Matérn class

$$k_{\mathrm{Matern}}(\mathbf{r}) = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\|\mathbf{r}\|_2}{l}\right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}\|\mathbf{r}\|_2}{l}\right), \tag{18a}$$

$$S_{\mathrm{Matern}}(\boldsymbol{\omega}) = \sigma_f^2 \frac{2^d \pi^{d/2}\Gamma(\nu + d/2)(2\nu)^{\nu}}{\Gamma(\nu) l^{2\nu}} \left(\frac{2\nu}{l^2} + \|\boldsymbol{\omega}\|_2^2\right)^{-(\nu+d/2)}, \tag{18b}$$

where K_ν is a modified Bessel function [35]. The smoothness of the process is increased with the parameter ν: in the limit $\nu \to \infty$ we recover the squared exponential covariance function.

Gaussian processes are also closely connected to classical spline smoothing [23] as well as other classical regularisation methods [21, 29] for inverse problems. Although the construction of the corresponding covariance function is hard (or impossible), it is still possible to construct the corresponding spectral density in many cases. With these spectral densities and the basis function method of Section 2.6, we can construct probabilistic versions of the classical regularisation methods as discussed in the next section.

2.5 Covariance functions arising from classical regularisation

Let us recall that a classical way to seek for solutions to inverse problems is via optimization of a functional of the form

$$\mathcal{J}[f] = \frac{1}{2\sigma^2}\sum_i (y_i - \mathcal{H}_{\mathbf{x},i} f(\mathbf{x}))^2 + \frac{1}{2\sigma_f^2}\int |\mathcal{L}f(\mathbf{x})|^2 \, d\mathbf{x}, \tag{19}$$

where $\mathcal{L}$ is a linear operator. This is equivalent to a Gaussian process regression problem, where the covariance operator is formally chosen to be $\mathcal{K} = [\mathcal{L}^* \mathcal{L}]^{-1}$. In (classical) Tikhonov regularisation we have $\mathcal{L} = \mathcal{I}$ (identity operator) which corresponds to penalising the norm of the solution. Another option is to penalise the Laplacian which gives $\mathcal{L} = \nabla^2$.

Although the kernel of this covariance operator is ill-defined with the classical choices of $\mathcal{L}$ and thus it is not possible to form the corresponding covariance function, we can still compute the corresponding spectral density function by computing the Fourier transform (15) of $\mathcal{L}^* \mathcal{L}$ and then inverting it to form the spectral density:

$$S(\boldsymbol{\omega}) = \frac{\sigma_f^2}{\mathcal{F}[\mathcal{L}^* \mathcal{L}]}. \tag{20}$$

In particular, the minimum norm or (classical) Tikhonov regularisation can be recovered by using a white noise prior which is given by the constant spectral density

$$S_{\text{Tikhonov}}(\boldsymbol{\omega}) = \sigma_f^2, \tag{21}$$

where σ_f is a scaling parameter. Another interesting case is the Laplacian operator based regularisation which corresponds to

$$S_{\text{Laplacian}}(\boldsymbol{\omega}) = \frac{\sigma_f^2}{\|\boldsymbol{\omega}\|_2^4}. \tag{22}$$

It is useful to note that the latter spectral density corresponds to a $l \to \infty$ limit of the Matérn covariance function with $\nu + d/2 = 2$ and the white noise to $l \to 0$ in either the SE or the Matérn covariance functions. The covariance functions corresponding to the above spectral densities would be degenerate, but this does not prevent us from using the spectral densities in the basis function expansion method described in Section 2.6 as the method only requires the availability of the spectral density.

2.6 Basis function expansion

To overcome the computational hazard described in Section 2.3, we consider the approximation method proposed in [44], which relies on the following truncated basis function expansion

$$k(\mathbf{x}, \mathbf{x}') \approx \sum_{i=1}^{m} S(\sqrt{\lambda_i}) \phi_i(\mathbf{x}) \phi_i(\mathbf{x}'), \tag{23}$$

where S denotes the spectral density of the covariance function, and m is the truncation number. The basis functions $\phi_i(\mathbf{x})$ and eigenvalues λ_i are obtained from the solution to the Laplace eigenvalue problem on the domain Ω

$$\begin{cases} -\Delta \phi_i(\mathbf{x}) = \lambda_i \phi_i(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \quad \phi_i(\mathbf{x}) = 0, & \mathbf{x} \in \partial\Omega. \end{cases} \tag{24}$$

In two dimensions with $\Omega = [-L_1, L_1] \times [-L_2, L_2]$ we introduce the positive integers $i_1 \leq m_1$ and $i_2 \leq m_2$. The number of basis functions is then $m = m_1 m_2$ and the solution to (24) is given by

$$\phi_i(\mathbf{x}) = \frac{1}{\sqrt{L_1 L_2}} \sin\big(\varphi_{i_1}(x_1 + L_1)\big) \sin\big(\varphi_{i_2}(x_2 + L_2)\big), \tag{25a}$$

$$\lambda_i = \varphi_{i_1}^2 + \varphi_{i_2}^2, \quad \varphi_{i_1} = \frac{\pi i_1}{2L_1}, \quad \varphi_{i_2} = \frac{\pi i_2}{2L_2}, \tag{25b}$$

where $i = i_1 + m_1(i_2 - 1)$. Let us now build the vector $\boldsymbol{\phi}_* \in \mathbb{R}^{m \times 1}$, the matrix $\Phi \in \mathbb{R}^{m \times M}$ and the diagonal matrix $\Lambda \in \mathbb{R}^{m \times m}$ as

$$(\boldsymbol{\phi}_*)_i = \phi_i(\mathbf{x}_*), \tag{26a}$$

$$\Phi_{ij} = \int_{-R}^{R} \phi_i(\mathbf{x}_j^0 + s\hat{\mathbf{u}}_j) ds, \tag{26b}$$

$$\Lambda_{ii} = S(\sqrt{\lambda_i}). \tag{26c}$$

The entries Φ_{ij} can be computed in closed form with details given in A. Now we substitute $Q \approx \Phi^\mathsf{T} \Lambda \Phi$ and $\mathbf{q}_* \approx \Phi^\mathsf{T} \Lambda \boldsymbol{\phi}_*$ to obtain

$$\mathbb{E}[f(\mathbf{x}_*) \mid \mathbf{y}] \approx \boldsymbol{\phi}_*^\mathsf{T} \Lambda \Phi (\Phi^\mathsf{T} \Lambda \Phi + \sigma^2 I)^{-1} \mathbf{y}, \tag{27a}$$

$$\mathbb{V}[f(\mathbf{x}_*) \mid \mathbf{y}] \approx \boldsymbol{\phi}_*^\mathsf{T} \Lambda \boldsymbol{\phi}_* - \boldsymbol{\phi}_*^\mathsf{T} \Lambda \Phi (\Phi^\mathsf{T} \Lambda \Phi + \sigma^2 I)^{-1} \Phi^\mathsf{T} \Lambda \boldsymbol{\phi}_*. \tag{27b}$$

When using the spectral densities corresponding to the classical regularisation methods in (21) and (22), the mean equation reduces to the classical solution (on the given basis). However, also for the classical regularisation methods we can compute the variance function which gives uncertainty estimate for the solution which in the classical formulation is not available. Furthermore, the hyperparameter estimation methods outlined in the next section provide principled means to estimate the parameters also in the classical regularisation methods.

3 Hyperparameter estimation

In this section, we will consider some methods for estimating the *hyperparameters*. The free parameters of the covariance function, for example, the parameters σ_f and l in the squared exponential covariance function, are together with the noise parameter σ referred to as the hyperparameters of the model. In this work, we employ a Bayesian approach to estimate the hyperparameters, and comparisons with standard parameter estimation methods such as L-curve and cross-validation methods are given as well.

3.1 Posterior distribution of hyperparameters

The marginal likelihood function corresponding to the model (12) is given as

$$p(\mathbf{y} \mid \sigma_f, l, \sigma) = \mathcal{N}(\mathbf{y} \mid \mathbf{0}, Q(\sigma_f, l) + \sigma^2 I), \tag{28}$$

where $Q(\sigma_f, l)$ is defined by (14a). The posterior distribution of parameters can now be written as follows:

$$p(\sigma_f, l, \sigma \mid \mathbf{y}) \propto p(\mathbf{y} \mid \sigma_f, l, \sigma) p(\sigma_f) p(l) p(\sigma), \tag{29}$$

where non-informative priors are used: $p(\sigma_f) \propto \frac{1}{\sigma_f}$, $p(l) \propto \frac{1}{l}$ and $p(\sigma) \propto \frac{1}{\sigma}$. The logarithm of (29) can be written as

$$\begin{aligned}\log p(\sigma_f, l, \sigma \mid \mathbf{y}) = \text{const.} - \frac{1}{2} \log\det(Q + \sigma^2 I) - \frac{1}{2}\mathbf{y}^\mathsf{T}(Q + \sigma^2 I)^{-1}\mathbf{y} \\ - \log\frac{1}{\sigma_f} - \log\frac{1}{l} - \log\frac{1}{\sigma}.\end{aligned} \tag{30}$$

Given the posterior distribution we have a wide selection of methods from statistics to estimate the parameters. One approach is to compute the maximum a posteriori (MAP) estimate of the parameters by using, for example, gradient-based optimisation methods [35]. However, using this kind of point estimate loses the uncertainty information of the hyperparameters and therefore in this article we use Markov chain Monte Carlo (MCMC) methods [5] which retain the information about the uncertainty in the final result.

3.2 Metropolis–Hastings sampling of hyperparameters

As discussed in the previous section, the statistical formulation of the inverse problem gives a posterior distribution of the hyperparameters $\boldsymbol{\varphi} = (\sigma_f, l, \sigma)$ as the solution rather than single estimates. The MCMC methods are capable of generating samples from the distribution. The Monte Carlo samples can then be used for computing the mean, the variance, or some other statistics of the posterior distribution [14]. In this work, we employ the Metropolis–Hastings algorithm to sample from the posterior distribution.

3.3 The L-curve method

One of the classical methods to obtain information about the optimum value for σ is the L-curve method [17], which operates by plotting the norm of the solution $\|f_\sigma(\mathbf{x})\|_2$ versus the residual norm $\|\mathcal{H}_{\mathbf{x},i} f_\sigma(\mathbf{x}) - y_i\|_2$. The associated L-curve is defined as the continous curve consisting of all the points $(\|\mathcal{H}_{\mathbf{x},i} f_\sigma(\mathbf{x}) - y_i\|_2, \|f_\sigma(\mathbf{x})\|_2)$ for $\sigma \in [0, \infty)$.

3.4 Cross-validation

As a comparison, we also consider to use methods of cross-validation (CV) for model selection. In k-fold CV, the data are partitioned into k disjoint sets $\mathbf{y}_j$, and at each round j of CV, the predictive likelihood of the set $\mathbf{y}_j$ is computed given the rest of the data $\mathbf{y}_{-j}$. These likelihoods are used to monitor the predictive performance of the model. This performance is used to estimate the generalisation error, and it can be used to carry out model selection [24, 35, 47].

The Bayesian CV estimate of the predictive fit with given parameters $\boldsymbol{\varphi}$ is

$$\mathrm{CV} = \sum_{j=1}^{n} \log p(\mathbf{y}_j \mid \mathbf{y}_{-j}, \boldsymbol{\varphi}), \tag{31}$$

where $p(\mathbf{y}_j \mid \mathbf{y}_{-j}, \boldsymbol{\varphi})$ is the predictive likelihood of the data $\mathbf{y}_j$ given the rest of the data. The best parameter values with respect to CV can be computed by enumerating the possible parameter values and selecting the one which gives the best fit in terms of CV.

4 Experimental results

In this section, we present numerical results using the GP model for limited x-ray tomography problems. All the computations were implemented in Matlab 9.4 (R2018a) and performed on an Intel Core i5 at 2.3 GHz and CPU 8GB 2133MHz LPDDR3 memory.

For both simulated data (see Section 4.1) and real data (see Section 4.2) we use $m = 10^4$ basis functions in (23). The measurements are obtained from the line integral of each x-ray over the attenuation coefficient of the measured objects. The measurements are taken for each direction (angle of view), and later they will be referred to as projections. The same number of rays in each direction is used. The computation of the hyperparameters is carried out using the Metropolis–Hastings algorithms with 5 000 samples, and the first 1 000 samples are thrown away (*burn-in* period). The reconstruction is computed by taking the conditional mean of the object estimate.

4.1 Simulated data: 2D Chest phantom

As for the simulated data, we use one slice of MATLAB's 3D Chest dataset [40] as a ground truth, f_{true}, which is shown in Figure 4(a). The size of the phantom is $N \times N$, with $N = 128$. The black region indicates zero values and lighter regions indicate higher attenuation function values. The measurements (i.e. sinogram) of the chest phantom are computed using the `radon` command in MATLAB and corrupted by additive white Gaussian noise with zero mean and 0.1 variance ($\sigma_{\text{true}} = 0.32$).

Several reconstructions of the chest phantom using different covariance functions, namely squared exponential (SE), Matérn, Laplacian, and Tikhonov, are presented. For the SE, Matérn, and Laplacian covariance functions, the parameters σ_f, l, and σ are estimated using the proposed method. We use $\nu = 1$ for the Matérn covariance. As for the Tikhonov covariance, it is not characterised by the length scale l, and hence only σ_f and σ are estimated. All the estimated parameters are reported in Table 4. Figure 3 presents the histograms of the 1-d marginal posterior distribution of each parameters using different covariance functions. The histograms show the distribution of the parameters samples in the Metropolis–Hastings samples. The results show that the σ_f estimate for SE and Matérn covariances is 0.12, while for Laplacian and Tikhonov, the estimates are 0.05 and 0.64. For Matérn, Laplacian, and Tikhonov covariance functions, the σ estimates are concentrated around the same values $0.34 - 0.39$ with standard deviation (SD) between $0.02 - 0.03$. These noise estimates are well-estimated the ground-truth noise, $\sigma_{\text{true}} = 0.32$, with the absolute error is between $0.02 - 0.07$. The estimate of the SE kernel appears to overestimate the noise, $\sigma = 0.60$. It is reported that the length-scale parameter, l, for Laplacian and SE covariance functions are concentrated in the same values, while for Matérn yields higher estimate, $l = 10.14$.

Figure 4(c)-(f) shows GP reconstructions of the 2D chest phantom using different covariance functions from 9 projections (uniformly spaced) out of 180° angle of view and 185 number of rays for each projection. The computation times for all numerical tests are reported in Table 1. The Metropolis–Hastings reconstruction shows longer computational time due to the need for generation of a large number of samples from the posterior distribution. However, the benefit of this algorithm is that it is easy to implement and it is reliable for sampling from high dimensional distributions.

The numerical test of the simulated data reconstructions is compared against figures of merit, namely:

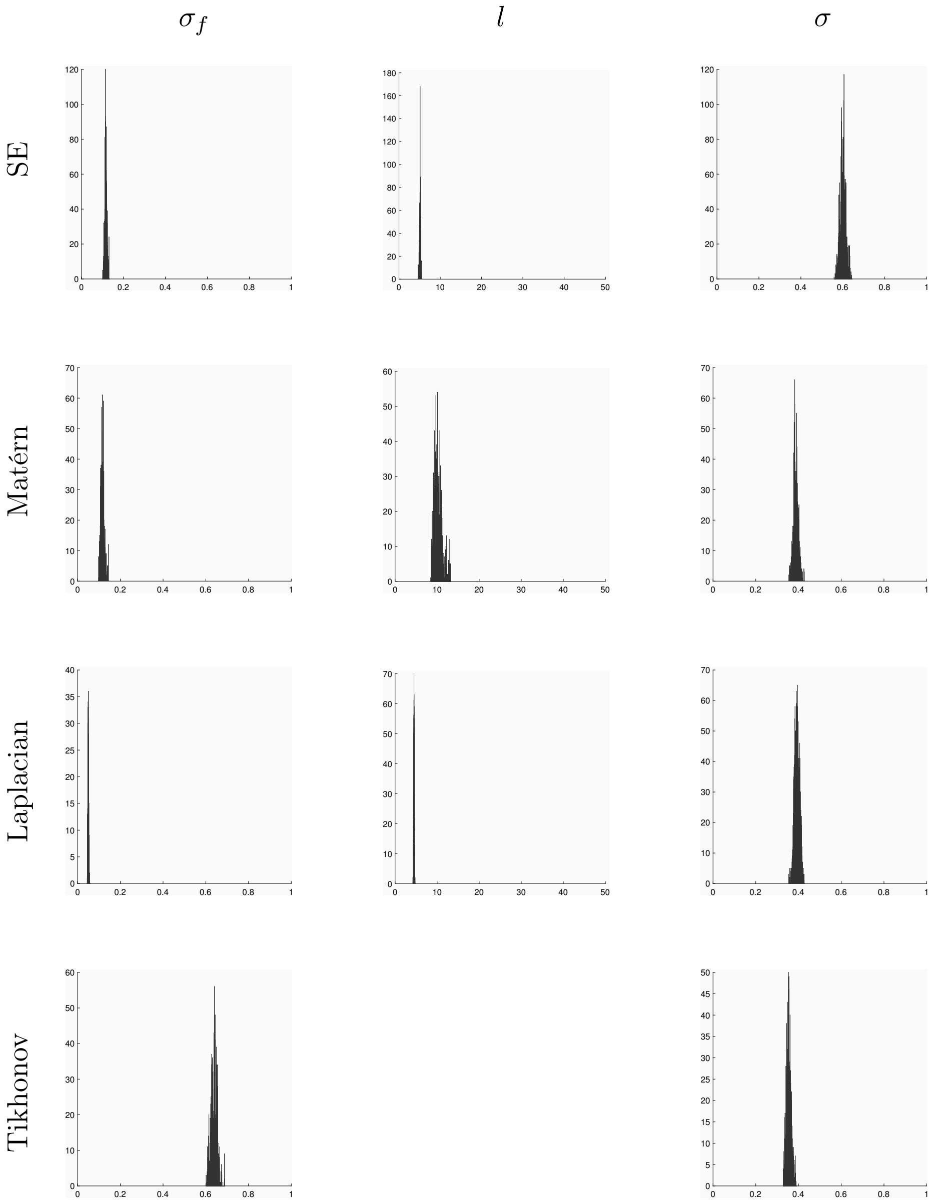


Figure 3: Histogram of the 1-d marginal distribution of the GP parameters. Left, middle and right columns are the marginal distribution for parameter σ_f, l and σ with corresponding covariance functions indicated in the vertical text in the left of the figure. The estimate of the parameter l is not available for Tikhonov covariance.

Table 1: Computation times of chest phantom (in seconds).

Target	FBP	SE	Matérn	Laplacian	Tikhonov
Chest phantom	0.5	11 210	9 676	9 615	9 615

- the relative error (RE)

  $$\frac{\|f_{\text{true}} - f_{\text{rec}}\|_2}{\|f_{\text{true}}\|_2},$$

  where f_{rec} is the image reconstruction, and

- the peak-signal-to-noise ratio (PSNR)

  $$10 \log_{10} \left(\frac{\text{peakval}^2}{\text{MSE}} \right),$$

  where peakval is the maximum possible value of the image and MSE is the mean square error between f_{true} and f_{rec},

as shown in Table 2.

In practice, image quality in CT depends on other parameters as well, such as image contrast, spatial resolution, and image noise [15]. These parameters can be evaluated when the CT device is equipped with CT numbers for various materials, high-resolution image is available, and statistical fluctuations of image noise which require several times of measurement to record random variations in detected x-ray intensity are acquired. However, in this work, the collected datasets are not supported by the aforementioned factors and they fall outside the scope of this paper. The results presented here are focusing on the implementation of a new algorithm to limited-data CT reconstruction and are reported as a preliminary study.

Reconstruction using a conventional method is computed as well with the built-in MATLAB function `iradon`, which uses the FBP to invert the Radon transform. It reconstructs a two-dimensional slice of the sample from the corresponding projections. The angles for which the projections are available are given as an argument to the function. Linear interpolation is applied during the backprojection and a Ram–Lak or ramp filter is used. The FBP reconstruction of the chest phantom is shown in Figure 4(b). For comparison, FBP reconstructions computed using some other filters are seen in Figure 5.

We also compared the results to the L-curve method and the CV:

- The L-curve method is applied to the Laplacian and the Tikhonov covariances and the L-curve plots from different values of parameter

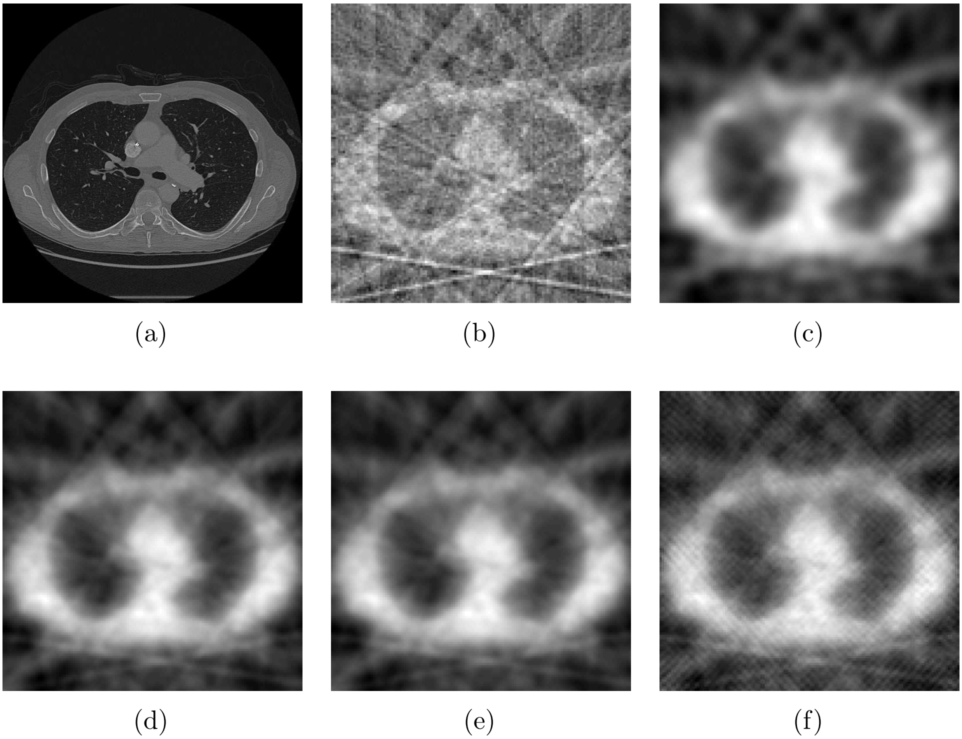
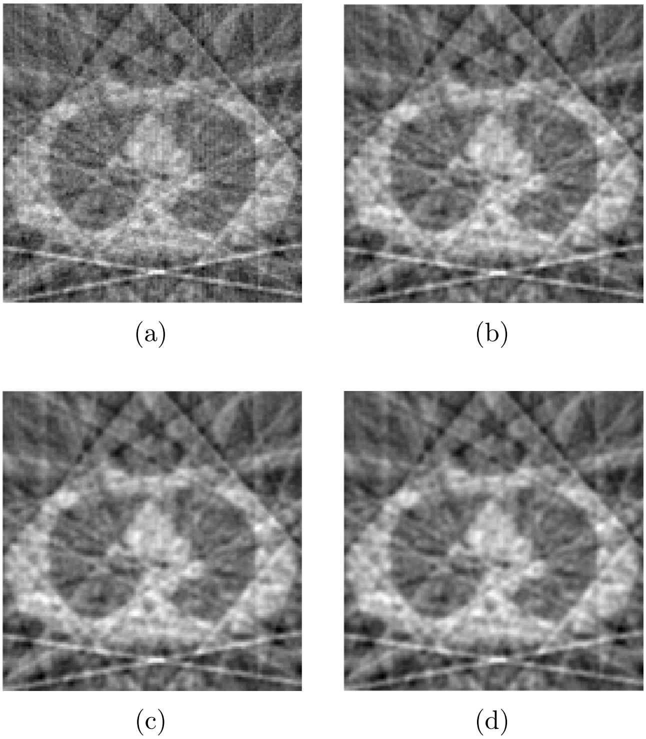


Figure 4: (a) A ground truth of 2D chest phantom. (b) Filtered backprojection reconstruction (Ram–Lak filter) from 9 projections. (c) GP reconstruction using SE covariance, (d) GP reconstruction using Matérn covariance, (e) GP reconstruction using Laplacian covariance, (f) GP reconstruction using Tikhonov covariance. The GP reconstructions are using 9 projections.

$10^{-1} \leq \sigma \leq 10$ for both covariances are shown in Figure 6. Both plots show that the corner of the L-curve is located in between $0.2 \leq \sigma \leq 1$.

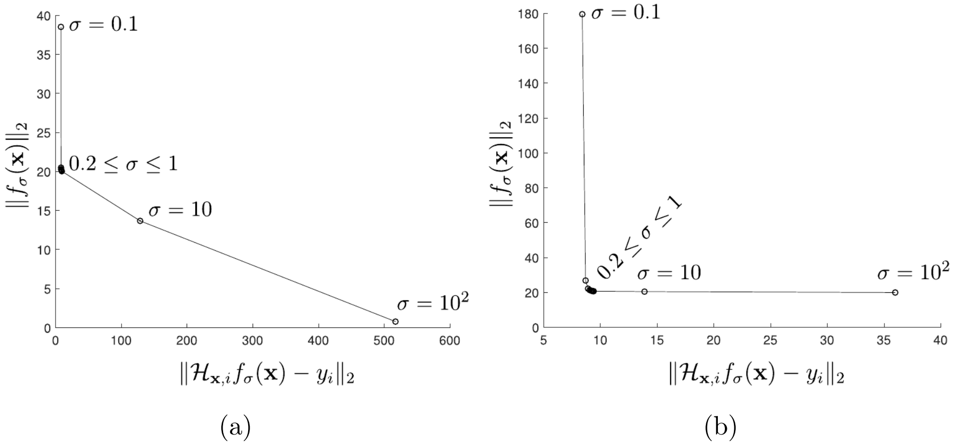
- The CV is tested for the Laplacian and Tikhonov covariances using point-wise evaluation of $10^{-2} \leq \sigma \leq 1$ and $10^{-2} \leq \sigma_f \leq 1$. For the Laplacian covariance, several points of length scale $1 \leq \ell \leq 100$ are tested as well. The minimum prediction error was obtained for $\sigma_f = 0.8$, $\sigma = 0.8$ and $\ell = 10$. For the Tikhonov covariance, the minimum prediction error was obtained for $\sigma = 0.5$ and $\sigma_f = 0.5$. The estimates of σ_f and σ for Laplacian are 0.8 and 0.5, respectively, and they give the same estimates for the Tikhonov covariance function. The estimates of σ for both kernels appear to overestimate the σ_{true}. The absolute error is in between $0.18 - 0.48$. The length-scale estimate from Laplacian covariance, $l = 10$, appears to close to the estimate in Matérn covariance.



(a) (b)

(c) (d)

Figure 5: Filtered backprojection reconstructions using (a) Shepp–Logan filter, (b) Cosine filter (c) Hamming filter, (d) Hann filter. Values of relative error (RE) are between $23.6-25.2$ and PSNR values are between $18.1-19.9\%$.



(a) (b)

Figure 6: The L-curve for (a) Tikhonov and (b) Laplacian covariance from the chest phantom reconstruction.

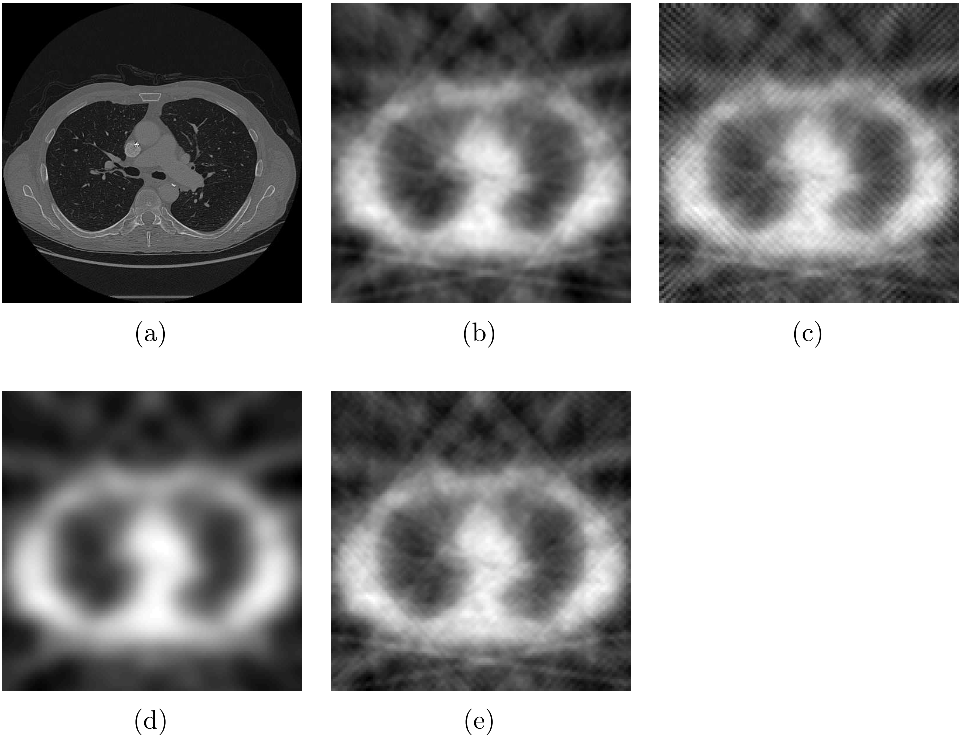


Figure 7: (a) A ground truth of 2D chest phantom. (b) & (c) reconstructions using L-curve parameter choice method with Laplacian (using $\sigma = 1$) and Tikhonov (using $\sigma = 0.2$) covariance functions, respectively. (d) & (e) reconstructions using CV with Laplacian and Tikhonov covariance functions, respectively

Image reconstructions for both L-curve and CV methods are shown in Figure 7.

4.2 Real data: Carved cheese

We now consider a real-world example using the tomographic x-ray data of a carved cheese slice measured with a custom-built CT device available at the University of Helsinki, Finland. The dataset is available online [1]. For a detailed documentation of the acquisition setup—including the specifications of the x-ray systems—see [6]. We use the downsampled sinogram with 140 rays and 15 projections from 360° angle of view. In the computations, the size of the target is set to 120×120.

Figure 8(c) shows the GP reconstruction (Matérn covariance function) of the cross section of the carved cheese slice using 15 projections (uniformly

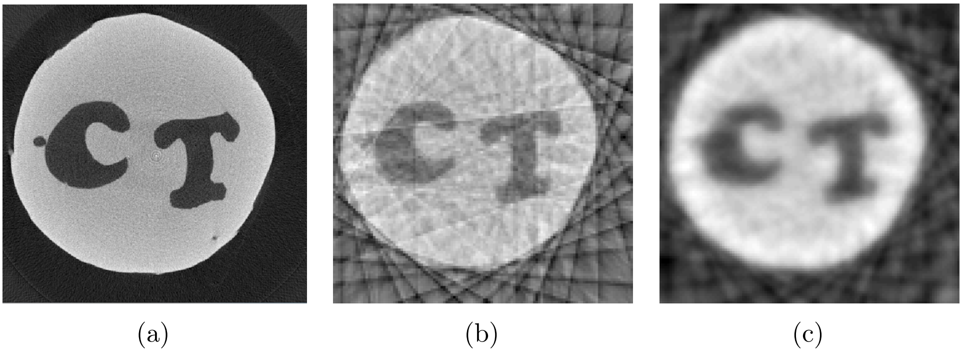


Figure 8: (a) FBP reconstruction (Ram–Lak filter) of the carved cheese using dense 360 projections. (b) Filtered backprojection reconstruction from 15 projections. (c) GP reconstruction using Matérn covariance from 15 projections.

spaced) out of 360° angle of view. For comparison, the FBP reconstruction is shown in Figure 8(b).

The computation times for the carved cheese are reported in Table 3.

4.3 Discussion

We have presented x-ray tomography reconstructions from both simulated and real data for limited projections (i.e. sparse sampling) using an approach based on the Gaussian process. However, other limited-data problems such as limited angle tomography could be explored as well. The quality of GP reconstructions using different covariance functions looks rather the same qualitatively. However, quantitatively, the reconstruction using Matérn covariance is the best one: it has the lowest RE 23.26% and the highest PSNR 22.76. PSNR describes the similarity of the original target with the reconstructed image (the higher value, the better of the reconstruction). Figures of merit estimates are not available for the real cheese data since there is no comparable ground truth. Nevertheless, the quality of the reconstruction can be observed qualitatively by comparing with the FBP reconstruction obtained with dense 360 projections from 360 degrees shown in Figure 8(a). The corresponding parameter estimates for the chest phantom and the cheese are reported in Table 4 and 5. For the chest phantom case, the estimate of parameter σ using Matérn, Laplacian and Tikhonov kernels tend to be close to the true value σ_{true}. As for the SE covariance, the standard deviation of noise is overestimated.

The reconstructions produced by the FBP benchmark algorithm using sparse projections are overwhelmed by streak artefacts due to the nature of

Table 2: Figures of merit for chest phantom reconstructions.

Method	RE (%)	PSNR
FBP (Ram–Lak filter)	25.86	18.44
GP-SE	29.41	21.76
GP-Matérn	23.26	22.76
GP-Laplacian	29.18	21.79
GP-Tikhonov	23.39	22.73
Lcurve-Laplacian	23.38	22.62
Lcurve-Tikhonov	23.26	22.63
CV-Laplacian	25.18	22.31
CV-Tikhonov	23.47	22.75

Table 3: Computation times of the carved cheese (in seconds).

Target	FBP	Matérn
Carved cheese	0.1	12 604

backprojection reconstruction, as shown in Figure 4(b) for the chest phantom and Figure 8(b) the for cheese target. The edges of the target are badly reconstructed. Due to the artefacts, especially for the chest phantom, it is difficult to distinguish the lighter region (which is assumed to be tissue) and the black region (air). The FBP reconstruction has the worst quality and it is confirmed in Table 2 that it has a high RE value (25.86%) and the lowest PSNR (18.44). FBP reconstructions computed with different filters are shown in Figure 5. However, there is no significant improvement in the images as it is clarified by the RE and PSNR values in the caption as well as by qualitative investigation. On the other hand, the GP reconstructions outperform the FBP algorithm in terms of image quality as reported in the figures of merit. The PSNR values of the GP-based reconstructions are higher than that of the FBP reconstruction. Nevertheless, in GP reconstructions, sharp boundaries are difficult to achieve due to the smoothness assumptions embedded in the model.

The GP prior clearly suppresses the artefacts in the reconstructions as shown in Figure 4(c) and 8(c). In Figure 4(c), the air and tissue region are recovered much better, since the prominent artefacts are much less. In Figure 8(c), the air region (outside the cheese and the C and T letters) are much sharper than in the FBP reconstruction. Overall, the results indicate

Table 4: The GP parameter estimates for the chest phantom. The estimates are calculated using the conditional mean, and the standard deviation (SD) values are also reported in parentheses.

Covariance function	σ_f (SD)	l (SD)	σ (SD)
SE	0.12 (0.04)	5.03 (0.03)	0.60 (0.02)
Matérn	0.12 (0.07)	10.14 (0.08)	0.34 (0.03)
Laplacian	0.05 (0.10)	4.49 (0.02)	0.39 (0.03)
Tikhonov	0.64 (0.02)	-	0.35 (0.03)

Table 5: Estimated GP parameters for the carved cheese using Matérn covariance function. The estimates are calculated using the conditional mean, and the standard deviation (SD) values are also reported in parentheses.

Covariance function	σ_f (SD)	l (SD)	σ (SD)
Matérn	0.012 (0.07)	11.00 (0.08)	0.02 (0.04)

that the image quality can be improved significantly by employing the GP method.

In Figure 7 the image reconstructions using L-curve and CV methods are presented. The quality of the reconstructions is reported in Table 2 as well. In these methods, finer point-wise evaluations might help to improve the quality of the reconstructions.

We emphasise that in the proposed GP-approach, some parameters in the prior is a part of the inference problem (see Equation (16)). Henceforth, we can avoid the difficulty in choosing the prior parameters. This problem corresponds to the classical regularisation methods, in which selecting the regularisation parameters is a very crucial step to produce a good reconstruction.

5 Conclusions

We have employed the Gaussian process with a hierarchical prior to computed tomography using limited projection data. The method was implemented to estimate the x-ray attenuation function from the measured data produced by the Radon transform. The performance has been tested on simulated and real data, with promising results shown. Unlike algorithms commonly used for the limited x-ray tomography problem that require manual tuning of prior parameters, the proposed GP method offers an easier set up as it takes into account the prior parameters as a part of the estimation. Henceforth, it constitutes a promising and user-friendly strategy.

The most important part of the GP model is the selection of the covariance function, since it stipulates the properties of the unknown function. As such, it also leaves most room for improvement. Considering the examples in Section 4, a common feature of the target functions is that they consist of a number of well-defined, separate regions. The function values are similar and thus highly correlated within the regions, while the correlation is low at the edges where rapid changes occur. This kind of behaviour is hard to capture with a stationary covariance function that models the correlation as completely dependent on the distance between the input locations. A non-stationary alternative is provided by, for example, the neural network covariance function, which is known for its ability to model functions with non-smooth features [35]. The basis function approximation method employed in this work is only applicable to stationary covariance functions, but other approximations can of course be considered.

Despite its success, the computational burden of the proposed algorithm is relatively high. To solve this problem, speed-up strategies are available, such as implementing parallelised GPU code, optimising the covariances of the sampling strategy, or by changing the MCMC algorithm to another one. Investigating finer resolution images and statistical records would also be interesting future research to evaluate other image quality parameters. Moreover, the proposed method can be applied to multidetector CT imaging [13, 28] as well as 3D CT problems using sparse data [33, 42].

6 Acknowledgements

Authors acknowledge support from the Academy of Finland (314474 and 313708). Furthermore, this research was financially supported by the Swedish Foundation for Strategic Research (SSF) via the project *ASSEMBLE* (contract number: RIT15-0012).

A Details on the computation of Φ

Here we derive the closed-form expression of the entries Φ_{ij} stated in (26b). We get that

$$
\begin{aligned}
\Phi_{ij} &= \int_{-R}^{R} \phi_i(\mathbf{x}_j^0 + s\hat{\mathbf{u}}_j)ds \\
&= \frac{1}{\sqrt{L_1L_2}} \int_{-R}^{R} \sin(\varphi_{i_1} r_j \cos\theta_j - \varphi_{i_1} s \sin\theta_j + \varphi_{i_1} L_1) \cdot \\
&\qquad\qquad\qquad \cdot \sin(\varphi_{i_2} r_j \sin\theta_j + \varphi_{i_2} s \cos\theta_j + \varphi_{i_2} L_2)ds \\
&= \frac{1}{\sqrt{L_1L_2}} \int_{-R}^{R} \sin(\alpha_{ij} s + \beta_{ij}) \sin(\gamma_{ij} s + \delta_{ij})ds \\
&= \frac{1}{2\sqrt{L_1L_1}} \int_{-R}^{R} \big[\cos((\alpha_{ij} - \gamma_{ij})s + \beta_{ij} - \delta_{ij}) \\
&\qquad\qquad\qquad - \cos((\alpha_{ij} + \gamma_{ij})s + \beta_{ij} + \delta_{ij})\big] ds \\
&= \frac{1}{2\sqrt{L_1L_1}} \Big[\frac{1}{\alpha_{ij} - \gamma_{ij}} \sin((\alpha_{ij} - \gamma_{ij})s + \beta_{ij} - \delta_{ij}) \\
&\qquad\qquad\qquad - \frac{1}{\alpha_{ij} + \gamma_{ij}} \sin((\alpha_{ij} + \gamma_{ij})s + \beta_{ij} + \delta_{ij})\Big]_{-R}^{R} \\
&= \frac{1}{2\sqrt{L_1L_1}} \Big(\frac{1}{\alpha_{ij} - \gamma_{ij}} \sin((\alpha_{ij} - \gamma_{ij})R + \beta_{ij} - \delta_{ij}) \\
&\qquad\qquad - \frac{1}{\alpha_{ij} + \gamma_{ij}} \sin((\alpha_{ij} + \gamma_{ij})R + \beta_{ij} + \delta_{ij}) \\
&\qquad\qquad - \frac{1}{\alpha_{ij} - \gamma_{ij}} \sin(-(\alpha_{ij} - \gamma_{ij})R + \beta_{ij} - \delta_{ij}) \\
&\qquad\qquad + \frac{1}{\alpha_{ij} + \gamma_{ij}} \sin(-(\alpha + \gamma_{ij})R + \beta_{ij} + \delta_{ij})\Big),
\end{aligned}
\tag{32}
$$

where

$$\alpha_{ij} = \varphi_{i_1} \sin\theta_j, \tag{33a}$$

$$\beta_{ij} = \varphi_{i_1} r_j \cos\theta_j + \varphi_{i_1} L_1, \tag{33b}$$

$$\gamma_{ij} = \varphi_{i_2} \cos\theta_j, \tag{33c}$$

$$\delta_{ij} = \varphi_{i_2} r_j \sin\theta_j + \varphi_{i_2} L_2. \tag{33d}$$

References

[1] https://www.fips.fi/datasetpage.php. Accessed: 2018-07-09.

[2] S. Akin and A. Kovscek. "Computed tomography in petroleum engineering research". In: *Geological Society, London, Special Publications* 215.1 (2003), pp. 23–38.

[3] C. Bouman and K. Sauer. "A generalized Gaussian image model for edge-preserving MAP estimation". In: *IEEE Transactions on Image Processing* 2.3 (July 1993), pp. 296–310.

[4] C. A. Bouman and K. Sauer. "A unified approach to statistical tomography using coordinate descent optimization". In: *IEEE Transactions on image processing* 5.3 (1996), pp. 480–492.

[5] S. Brooks, A. Gelman, G. Jones and X.-L. Meng. *Handbook of markov chain monte carlo*. CRC press, 2011.

[6] T. A. Bubba, M. Juvonen, J. Lehtonen, M. März, A. Meaney, Z. Purisha and S. Siltanen. "Tomographic X-ray data of carved cheese". In: *arXiv preprint arXiv:1705.05732* (2017).

[7] T. M. Buzug. *Computed tomography: from photon statistics to modern cone-beam CT*. Springer Science & Business Media, 2008.

[8] L. Cartz. *Nondestructive testing*. ASM International, 1995.

[9] A. M. Cormack. "Representation of a function by its line integrals, with some radiological applications". In: *Journal of Applied physics* 34.9 (1963), pp. 2722–2727.

[10] A. M. Cormack. "Representation of a function by its line integrals, with some radiological applications. II". In: *Journal of Applied Physics* 35.10 (1964), pp. 2908–2913.

[11] L. De Chiffre, S. Carmignato, J.-P. Kruth, R. Schmitt and A. Weckenmann. "Industrial applications of computed tomography". In: *CIRP Annals-Manufacturing Technology* 63.2 (2014), pp. 655–677.

[12] D. Fanelli and O. Öktem. "Electron tomography: a short overview with an emphasis on the absorption potential model for the forward problem". In: *Inverse Problems* 24.1 (2008), p. 013001.

[13] T. G. Flohr, S. Schaller, K. Stierstorfer, H. Bruder, B. M. Ohnesorge and U. J. Schoepf. "Multi–detector row CT systems and image-reconstruction techniques". In: *Radiology* 235.3 (2005), pp. 756–773.

[14] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari and D. B. Rubin. *Bayesian Data Analysis*. Third. Chapman and Hall/CRC, 2013.

[15] L. W. Goldman. “Principles of CT: radiation dose and image quality”. In: *Journal of nuclear medicine technology* 35.4 (2007), pp. 213–225.

[16] H. Haario, A. Kallonen, M. Laine, E. Niemi, Z. Purisha and S. Siltanen. “Shape recovery for sparse-data tomography”. In: *Mathematical Methods in the Applied Sciences* 40.18 (2017), pp. 6649–6669.

[17] P. C. Hansen. “Analysis of discrete ill-posed problems by means of the L-curve”. In: *SIAM review* 34.4 (1992), pp. 561–580.

[18] J. Hendriks, A. Gregg, C. Wensrich and W. Wills. “Implementation of Traction Constraints in Bragg-edge Neutron Transmission Strain Tomography”. In: *arXiv preprint arXiv:1805.09760* (2018).

[19] G. Herman. “Image Reconstruction From Projections”. In: *Real-Time Imaging* 1.1 (1995), pp. 3–18. ISSN: 1077-2014.

[20] C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. “Probabilistic modelling and reconstruction of strain”. In: *arXiv preprint arXiv:1802.03636* (2018).

[21] J. Kaipio and E. Somersalo. *Statistical and computational inverse problems*. Vol. 160. Springer Science & Business Media, 2006.

[22] A. C. Kak and M. Slaney. *Principles of computerized tomographic imaging*. IEEE press New York, 1988.

[23] G. S. Kimeldorf and G. Wahba. “A Correspondence Between Bayesian Estimation on Stochastic Processes and Smoothing by Splines”. In: *The Annals of Mathematical Statistics* 41.2 (1970), pp. 495–502.

[24] R. Kohavi et al. “A study of cross-validation and bootstrap for accuracy estimation and model selection”. In: *Ijcai*. Vol. 14. 2. Montreal, Canada. 1995, pp. 1137–1145.

[25] V. Kolehmainen, S. Siltanen, S. Järvenpää, J. P. Kaipio, P. Koistinen, M. Lassas, J. Pirttilä and E. Somersalo. “Statistical inversion for medical x-ray tomography with few radiographs: II. Application to dental radiology”. In: *Physics in Medicine & Biology* 48.10 (2003), p. 1465.

[26] P. Kuchment. *The Radon transform and medical imaging*. SIAM, 2013.

[27] D. Li, J. Svensson, H. Thomsen, F. Medina, A. Werner and R. Wolf. “Bayesian soft X-ray tomography using non-stationary Gaussian Processes”. In: *Review of Scientific Instruments* 84.8 (2013), p. 083506.

[28] M. R. K. Mookiah, K. Subburaj, K. Mei, F. K. Kopp, J. Kaesmacher, P. M. Jungmann, P. Foehr, P. B. Noel, J. S. Kirschke and T. Baum. “Multidetector computed tomography imaging: effect of sparse sampling and iterative reconstruction on trabecular bone microstructure”. In: *Journal of computer assisted tomography* 42.3 (2018), pp. 441–447.

[29] J. L. Mueller and S. Siltanen. *Linear and nonlinear inverse problems with practical applications.* Vol. 10. Siam, 2012.

[30] National Research Council. *Mathematics and Physics of Emerging Biomedical Imaging.* Washington, DC: The National Academies Press, 1996.

[31] F. Natterer. *The mathematics of computerized tomography.* Vol. 32. Siam, 1986.

[32] L. T. Niklason, B. T. Christian, L. E. Niklason, D. B. Kopans, D. E. Castleberry, B. Opsahl-Ong, C. E. Landberg, P. J. Slanetz, A. A. Giardino, R. Moore et al. "Digital tomosynthesis in breast imaging". In: *Radiology* 205.2 (1997), pp. 399–406.

[33] Z. Purisha, S. S. Karhula, J. H. Ketola, J. Rimpeläinen, M. T. Nieminen, S. Saarakkala, H. Kröger and S. Siltanen. "An Automatic Regularization Method: An Application for 3-D X-Ray Micro-CT Reconstruction Using Sparse Data". In: *IEEE transactions on medical imaging* 38.2 (2018), pp. 417–425.

[34] M. Rantala, S. Vanska, S. Jarvenpaa, M. Kalke, M. Lassas, J. Moberg and S. Siltanen. "Wavelet-based reconstruction for limited-angle X-ray tomography". In: *IEEE transactions on medical imaging* 25.2 (2006), pp. 210–217.

[35] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning.* MIT press, Cambridge, MA, 2006.

[36] N. Riis, J. Frøsig, Y. Dong and P. Hansen. "Limited-data x-ray CT for underwater pipeline inspection". In: *Inverse Problems* 34.3 (2018), p. 034002.

[37] J. Sachs and K. Sauer. "3D Reconstruction from sparse Radiographic Data". In: *Discrete Tomography.* Springer, 1999, pp. 363–383.

[38] S. Särkkä. "Linear Operators and Stochastic Partial Differential Equations in Gaussian Process Regression". In: *Proceedings of ICANN.* 2011.

[39] K. Sauer, J. Sachs and C. Klifa. "Bayesian estimation of 3-D objects from few radiographs". In: *IEEE Transactions on Nuclear Science* 41.5 (1994), pp. 1780–1790.

[40] *Segment Lungs from 3-D Chest Scan and Calculate Lung Volume.* `https://se.mathworks.com/help/images/segment-lungs-from-3-d-chest-mri-data.html`. Accessed: 2018-07-09.

[41] L. A. Shepp and J. Kruskal. "Computerized tomography: the new medical X-ray technology". In: *American Mathematical Monthly* 85 (1978), pp. 420–439.

[42] E. Y. Sidky and X. Pan. “Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization”. In: *Physics in Medicine & Biology* 53.17 (2008), p. 4777.

[43] S. Siltanen, V. Kolehmainen, S. Järvenpää, J. Kaipio, P. Koistinen, M. Lassas, J. Pirttilä and E. Somersalo. “Statistical inversion for medical x-ray tomography with few radiographs: I. General theory”. In: *Physics in Medicine & Biology* 48.10 (2003), p. 1437.

[44] A. Solin and S. Särkkä. “Hilbert space methods for reduced-rank Gaussian process regression”. In: *Statistics and Computing* (Aug. 2019).

[45] J. Svensson. *Non-parametric tomography using Gaussian processes.* EFDA, 2011.

[46] A. Tarantola. *Inverse problem theory and methods for model parameter estimation.* Vol. 89. siam, 2005.

[47] A. Vehtari, A. Gelman and J. Gabry. “Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC”. In: *Statistics and Computing* 27.5 (2017), pp. 1413–1432.

[48] T. Wu, A. Stewart, M. Stanton, T. McCauley, W. Phillips, D. B. Kopans, R. H. Moore, J. W. Eberhard, B. Opsahl-Ong, L. Niklason et al. “Tomographic mammography using a limited number of low-dose cone-beam projection images”. In: *Medical physics* 30.3 (2003), pp. 365–380.

[49] Y. Zhang, H.-P. Chan, B. Sahiner, J. Wei, M. M. Goodsitt, L. M. Hadjiiski, J. Ge and C. Zhou. “A comparative study of limited-angle cone-beam reconstruction methods for breast tomosynthesis”. In: *Medical physics* 33.10 (2006), pp. 3781–3795.

Paper IV

Title

Deep kernel learning for integral measurements.

Authors

Carl Jidling, Johannes Hendriks, Thomas B. Schön, and Adrian Wills.

Statement of contribution

The idea to this paper is credited Johannes Hendriks. Carl Jidling undertook most of the implementation and the majority of the writing, with essential contributions from Hendriks. Fruitful comments and feedback were provided by the remaining authors.

Edited version of

C. Jidling, T. B. Schön, J. Hendriks and A. Wills. *Deep kernel learning for integral measurements.* Tech. rep. arXiv:1909.01844, Sept. 2019.

Deep kernel learning for integral measurements

Abstract

Deep kernel learning refers to a Gaussian process that incorporates neural networks to improve the modelling of complex functions. We present a method that makes this approach feasible for problems where the data consists of line integral measurements of the target function. The performance is illustrated on computed tomography reconstruction examples.

1 Introduction

The Gaussian process (GP) [33] is a powerful regression tool that has been successfully applied to problems within many different fields. Encoding a broad class of non-linear functions, a key feature of the GP is the ability to adapt its complexity with the size of the data set while keeping a constant number of free hyperparameters; this is referred to as flexibility. The performance and accuracy of the GP is, however, in no small part determined by the model assumptions embedded in the associated covariance function.

The most common covariance functions are stationary, which means that the modelled correlation between two function values is dependent purely on the distance between their corresponding input locations. A notable member of this class is the *squared exponential* covariance function, which is widely employed mainly due to its ease of implementation.

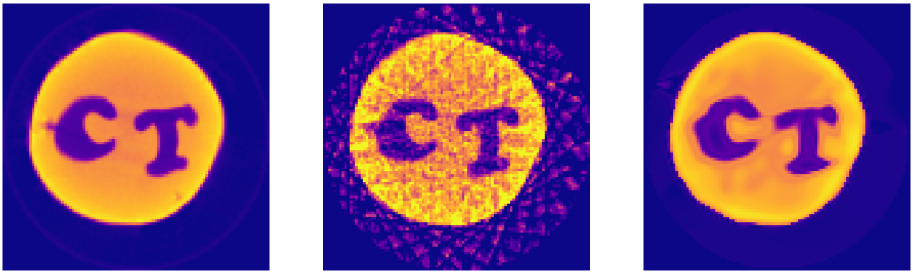


Figure 1: The proposed method applied to computed tomography reconstruction from X-ray data. The following example is from the scanning of a carved cheese. Left: reference reconstruction of high accuracy. Middle: reconstruction using filtered back projection, a commonly used method in practise. Right: reconstruction using the proposed method.

Although stationary covariance functions are intuitive and rather realistic for many functions, this choice causes severe problems if the target function contains non-smooth features, such as rapid, step-like changes. A way of meeting this challenge is to use a non-stationary covariance function. In particular, the *neural network* covariance function [28] is known for its ability to capture non-stationary features. However, it is harder to implement and extend beyond point measurement models (direct observations of the target function), e.g. extension to integral measurements.

An alternative non-stationary construction is obtained by warping the inputs to a stationary covariance function through a non-linear mapping [33]. Letting this mapping be a neural network, we obtain the framework known as *manifold Gaussian processes* [5] or *deep kernel learning* [54], which has been demonstrated on point measurements with promising potential.

The practical procedure of this approach becomes more challenging when we consider more advanced measurement models. In this work, we propose a method that allows for the application of deep kernel learning to problems where the measured data is expressed as line integrals of the target function, which arise for instance within X-ray computed tomography (CT) [7, 21, 41] and strain field estimation [19, 27, 50]. This is a non-trivial extension considering that a straightforward naive implementation requires numerical double integral computations in a number that scales quadratically with the size of the data set. Also, the training procedure is challenging since the cost function contains many local minima and the convergence is dependent upon a suitable initialisation.

To overcome these obstacles we approximate the GP with a Hilbert space basis function expansion [46], and so reduce the numerical computation to *single integrals* in a number that scales *linearly* with the size of the data set. Furthermore, by exploiting the model setup we pre-train the neural network used in the covariance function to provide a customised initialisation for the remaining joint training when incorporated within the deep kernel model. This significantly improves the end result and overall robustness.

The potential is demonstrated on simulated and real-data CT reconstruction problems, with promising results shown. An illustration is seen in Figure 1.

2 Background on the model

In this section, we briefly introduce the model background that forms the foundation of our developments; integral measurements in GPs and the deep kernel learning formulation.

2.1 Gaussian processes with integral measurement

The GP can be seen as a distribution over functions where any finite set of function values has a joint Gaussian distribution. Formally we write

$$f(\mathbf{x}) \sim \mathcal{GP}\left(0,\ k(\mathbf{x}, \mathbf{x}')\right), \tag{1}$$

to denote that the function $f(\mathbf{x}) : \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}$ is modelled as a zero-mean GP with covariance function $k(\mathbf{x}, \mathbf{x}') : \mathbb{R}^{D_\mathbf{x}} \times \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}$, and $\mathbf{x} = [x_1, \ldots, x_{D_\mathbf{x}}]^\mathsf{T} \in \mathbb{R}^{D_\mathbf{x}}$.

An important and very useful property of the GP is that it is closed under linear functional evaluations [13, 24, 29, 33]. This means that when a linear functional $\mathcal{L}$ is acting on a GP, the result is also a GP. Hence, it holds that

$$\mathcal{L}f(\mathbf{x}) \sim \mathcal{GP}\left(0,\ \mathcal{L}\mathcal{L}'k(\mathbf{x}, \mathbf{x}')\right), \tag{2}$$

where $\mathcal{L}'$ denotes the functional acting on the second argument of $k(\mathbf{x}, \mathbf{x}')$. Considering line integrals along straight line segments, we define the functional as

$$\mathcal{L}_i f(\mathbf{x}) \triangleq \int_{-r_i}^{r_i} f(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i) ds, \tag{3}$$

where $\mathbf{x}_i^0$ denotes the centre of the line, $\hat{\mathbf{n}}_i$ is a unit vector specifying the direction, r_i is the integration radius (half the line length) and the index i refer to the i^{th} data point. The corresponding covariance transformation yields the double integral

$$\mathcal{L}_i\mathcal{L}_j' k(\mathbf{x}, \mathbf{x}') = \int_{-r_i}^{r_i} \int_{-r_j}^{r_j} k(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i, \mathbf{x}_j^0 + s'\hat{\mathbf{n}}_j) ds ds', \tag{4}$$

which gives the covariance between measurement i and j. Let the measurements be stored in the vector $\mathbf{y} = [y_1, \ldots, y_N]^\mathsf{T}$ with

$$y_i = \mathcal{L}_i f(\mathbf{x}) + \varepsilon_i, \tag{5}$$

where the noise $\varepsilon_i \sim \mathcal{N}\left(0,\ \sigma^2\right)$. Furthermore, we are interested in the prediction $\mathbf{f}_* = [f(\mathbf{x}_{*1}) \cdots f(\mathbf{x}_{*N_*})]$, the function values at a set of unseen input locations $\{\mathbf{x}_{*i}\}_{i=1}^{N_*}$. Since linear transformations preserve Gaussianity, $\mathbf{y}$ and $\mathbf{f}_*$ have a joint Gaussian distribution:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathscr{L} + \sigma^2 I & \mathscr{L}_* \\ \mathscr{L}_*^\mathsf{T} & K_{**} \end{bmatrix} \right), \tag{6}$$

where $\mathscr{L}_{ij} = \mathcal{L}_i \mathcal{L}'_j k(\mathbf{x}, \mathbf{x}')$, $(\mathscr{L}_*)_{ij} = \mathcal{L}_i k(\mathbf{x}, \mathbf{x}_{*j})$, and $(K_{**})_{ij} = k(\mathbf{x}_{*i}, \mathbf{x}_{*j})$. The predictive expressions given this joint prior are given by

$$\mathbb{E}[\mathbf{f}_*|\mathbf{y}] = \mathscr{L}_*^{\mathsf{T}} \mathscr{L}^{-1} \mathbf{y}, \tag{7a}$$

$$\mathrm{Cov}\,[\mathbf{f}_*|\mathbf{y}] = K_{**} - \mathscr{L}_*^{\mathsf{T}} \mathscr{L}^{-1} \mathscr{L}_*. \tag{7b}$$

Thus, we can make predictions of the function values $\mathbf{f}_*$ purely based on line integral data; note that integration is a *conservative* functional, meaning that all information about the function is preserved under its evaluation (as opposed to e.g. differentiation). The challenging part here lies in the computation of the integral expressions, especially the double integrals (4) in $\mathscr{L}$.

2.2 Deep kernel learning

The most crucial part of Gaussian process modelling is the selection of the covariance function $k(\mathbf{x}, \mathbf{x}')$, since it stipulates the basic behaviour of the target function $f(\mathbf{x})$. The most common covariance functions are stationary such that $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$. Prominent members of this class include the Matérn family [47], the so-called spectral mixture kernels [52], and the popular squared exponential covariance function:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left[-\frac{1}{2}\sum_{k=1}^{D_\mathbf{x}} l_k^{-2}(x_k - x'_k)^2\right], \tag{8}$$

parameterised by the magnitude parameter σ_f and the lengthscales l_k, which impact how quickly the function may change.

In order to extend the expressiveness of stationary covariance functions, non-stationarity can be introduced by transforming the inputs through a non-linear mapping $\mathbf{u}(\cdot) : \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}^{D_\mathbf{u}}$ to form $k(\mathbf{u}(\mathbf{x}), \mathbf{u}(\mathbf{x}'))$ [33]. The dimension $D_\mathbf{u}$ of $\mathbf{u}(\cdot)$ can be chosen arbitrarily, and may therefore differ from the dimension $D_\mathbf{x}$ of $\mathbf{x}$.

Using this construction in the modelling of complex functions with limited prior knowledge, we need $\mathbf{u}(\cdot)$ to encode a general class of functions that can be learnt from data. A natural choice is to let $\mathbf{u}(\cdot)$ be described by a neural network. This is the idea behind manifold GPs [5] and deep kernel learning [54]. The $D_\mathbf{u}$ latent outputs $u_j(\cdot)$ are either completely independent, or they are different outputs of the same network, see Figure 2 for an illustration. The intuition is that the neural network does not have to learn the complete function $f(\mathbf{x})$, but only identify its discontinuities while for the remaining part the model can rely upon the regression capabilities of the GP.

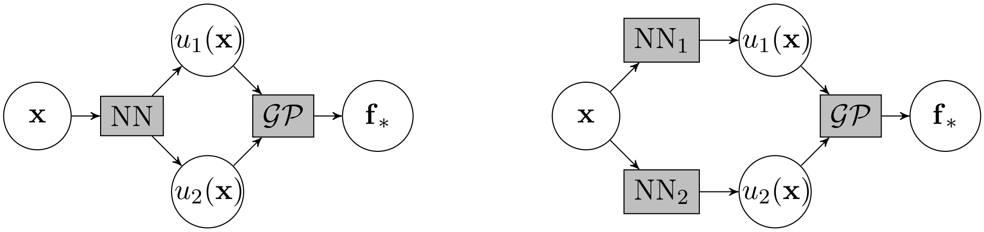


Figure 2: Illustration of deep kernel learning and two different constructions for the latent mapping $\mathbf{u}(\mathbf{x}) = [u_1(\mathbf{x})\;\; u_2(\mathbf{x})]^\mathsf{T}$. Left: the components $u_1(\mathbf{x})$ and $u_2(\mathbf{x})$ are different outputs of the same neural network. Right: the components are independent outputs of two different networks.

3 Deep kernel learning with integral measurements

Our aim in this work is to combine the GPs ability of incorporating line integral measurement with the neural network warping to form a method that is practically feasible beyond one-dimensional problems.

3.1 Basis function expansion

To reduce the computational load, we make use of a Hilbert space approximation method for GP regression [46]. In this approach a stationary covariance function is approximated by the following finite sum:

$$k(\mathbf{u}, \mathbf{u}') \approx \sum_{j=1}^{m} S(\mathbf{c}_j)\phi_j(\mathbf{u})\phi_j(\mathbf{u}'), \tag{9}$$

where $S(\cdot)$ denotes the spectral density of the covariance function. The basis functions $\{\phi_j(\cdot)\}_{j=1}^m$ with corresponding eigenvalues $\{\lambda_j\}_{j=1}^m$ are obtained from the Laplace eigenvalue problem

$$\begin{cases} -\Delta\phi_j(\mathbf{u}) = \lambda_j\phi_j(\mathbf{u}), & \mathbf{u} \in \Omega, \\ \quad\;\; \phi_j(\mathbf{u}) = 0, & \mathbf{u} \in \partial\Omega, \end{cases} \tag{10}$$

where $\Omega = [L_1, L_1] \times \cdots \times [L_{D_\mathbf{u}}, L_{D_\mathbf{u}}]$ is a generalised rectangular domain, and Δ denotes the Laplace operator. Here a Dirichlet boundary condition is used, but it does not affect the GP solution if L_k is chosen carefully, which is discussed in the supplementary material; for problems where the boundary conditions are explicitly specified, more advanced formulations are possible

[45]. The solution to (10) is given by

$$\phi_j(\mathbf{u}) = \prod_{k=1}^{D_\mathbf{u}} L_k^{-1/2} \sin\left[c_{kj}(u_k + L_k)\right], \qquad c_{kj} = \frac{j_k \pi}{2L_k}, \qquad \lambda_j = \sum_{k=1}^{D_\mathbf{u}} c_{kj}^2. \tag{11}$$

The notation denotes that basis function j has index $j_k \in [1, \ldots, \tilde{m}]$ in direction k. Using $\tilde{m}$ basis functions in each direction, we get a total number of $m = \tilde{m}^{D_\mathbf{u}}$. Furthermore, we have introduced $\mathbf{c}_j = [c_{1j}, \ldots, c_{D_\mathbf{u}j}]$ for the input to $S(\cdot)$ in (9).

With the network warping included, the matrix $\mathscr{L}$ in (6) is approximated as $\mathscr{L} \approx \Phi\Lambda\Phi^\mathsf{T}$ where

$$\Phi_{ij} = \mathcal{L}_i \phi_j(\mathbf{u}(\mathbf{x})), \qquad \Lambda_{jj} = S(\mathbf{c}_j). \tag{12}$$

Using the matrix inversion lemma, the approximate versions of the predictive expressions (7) are reformulated for more efficient computations provided that $m < N$. However, the main advantage of this method is the separation of the inputs $\mathbf{x}$ and $\mathbf{x}'$ in the basis function product. A consequence of this separation is that the double integral computations required to build $\mathscr{L}$ *reduce to single* integral computations of the form

$$\Phi_{ij} = \int_{-r_i}^{r_i} \prod_{k=1}^{D_\mathbf{u}} L_k^{-1/2} \sin\left[c_{kj}(u_k(\mathbf{x}_i^0 + s\hat{\mathbf{n}}_i) + L_k)\right] ds. \tag{13}$$

This integral can not be computed in closed form due to the non-linearity $\mathbf{u}(\cdot)$, but numerical integration is nevertheless significantly less demanding in one dimension than it is in two (compare with expression (4)). For instance, we can use a direct scheme such as the composite Simpson's 1/3 rule of integration [6].

3.2 Training the model

The model as formulated above contains the free hyperparameters $\theta = [\theta_\mathrm{k}^\mathsf{T} \ \theta_\mathbf{u}^\mathsf{T}]^\mathsf{T}$, which we separate with respect to the covariance function (θ_k) and the neural network ($\theta_\mathbf{u}$), respectively. As for the squared exponential covariance function (8), we have $\theta_\mathrm{k} = \{\sigma_f, \{l_k\}, \sigma\}$, including the standard deviation σ of the noise. There are different cost function options available for training θ, among which two common ones are the marginal likelihood (ML) and leave-one-out cross-validation (LOO-CV) [33]. In our experience, these two methods have shown a similar performance. Regardless of choice, the numerical robustness of the computations is improved using the *QR*-factorisation with details given in the supplementary material.

An important aspect of the training procedure is the parameter initialisation in the optimisation routine. This is a non-trivial challenge as the total number of parameters is large due to the neural network. Moreover, the initialisation typically has big impact on the resulting optimisation performance and hence also on the quality of the final prediction. For challenging problems with complex two-dimensional functions, pre-training of the neural network as described below has shown to have a crucial impact on the convergence.

To obtain a satisfying initial guess, let us take a moment to reflect on what we want $\mathbf{u}(\cdot)$ to achieve. The reason for introducing this mapping is that a stationary covariance function always assigns high correlation to function values at closely located inputs. In other words, if the distance $|\mathbf{x} - \mathbf{x}'|$ is small, then $f(\mathbf{x})$ and $f(\mathbf{x}')$ are assumed to be similar. In regions of rapid changes and discontinuities, this assumption fails drastically. We concretise this by considering inference of a one-dimensional step function while using a scalar latent mapping $u(\cdot)$. Two points x_0 and x_1 located just before and just after the step differ significantly in their function values $f(x_0)$ and $f(x_1)$. Therefore, we want to train $u(\cdot)$ such that $u(x_0)$ and $u(x_1)$ become clearly separated, and hence make $f(x_0)$ and $f(x_1)$ weakly correlated. Considering the remaining parts of the step function, it consists of two constant regions where we also want $u(\cdot)$ to be constant for maximum correlation.

Extending this reasoning, it is easy to imagine several different mappings that would yield ideal correlation assignments by the stationary covariance function, with the essential feature being identification of discontinuities and distinguishing between points that are separated by them. An intuitive ideal mapping is $\mathbf{u}(\mathbf{x})$ being equal to the target function, since this choice assigns maximum correlation to identical function values. Although other ideal mappings might be less complex and more robust, this one is natural in lack of other prior information. Thus, our proposed pre-training aims at finding a latent mapping that is a reasonable approximation of the true function.

To begin with, we restrict ourselves to the case $D_{\mathbf{u}} = 1$ for now where we denote $\mathbf{u}(\mathbf{x})$ with $u(\mathbf{x})$. Although the model might be more expressive with several latent outputs, the computational load increases since it requires more basis functions; the number scales exponentially with $D_{\mathbf{u}}$ for a retained frequency resolution.

For pre-training the neural network parameters θ_u, we suggest the choice

$$\theta_u = \underset{\theta_u}{\operatorname{argmin}} \frac{1}{N_t} \sum_{i=1}^{N_t} (f_t(\mathbf{x}_i) - u(\mathbf{x}_i))^2, \tag{14}$$

where $\{\mathbf{x}_t\}_{i=1}^{N_t}$ is a set of N_t points in the domain of interest, and $f_t(\cdot)$ denotes the mean prediction obtained from standard GP reconstruction. As stated,

Algorithm 1 Deep kernel learning with line integral measurements

```
Input: Data set $\{y, \mathbf{x}^0, r, \hat{\mathbf{n}}\}_{i=1}^N$
Output: $\mathbb{E}[\mathbf{f}_*|\mathbf{y}], \mathrm{Cov}\,[\mathbf{f}_*|\mathbf{y}]$
1. Pre-train the neural network $u(\mathbf{x})$ using (14).
2. Train the extended model.
3. Compute the mean prediction $\mathbb{E}[\mathbf{f}_*|\mathbf{y}]$ and the covariance $\mathrm{Cov}\,[\mathbf{f}_*|\mathbf{y}]$ using (7).
```

this approach does not directly generalise to the case $D_{\mathbf{u}} > 1$. However, it can still be employed in such constructions. For instance, one could combine a pre-trained neural network mapping $u(\mathbf{x})$ with the mappings $u_k(\mathbf{x}) = x_k$ (whereby the original inputs are also used).

A natural question following this pre-training is why we need the neural network; we could as well remove this intermediate step and fix $u(\cdot)$ to be the output of the standard GP. However, a standard GP prediction is likely to contain undesired artefacts for problems with discontinuous features, and these artefacts may have negative impact when propagated through to another GP. With a neural network warping, the joint training is capable of eliminating or at least drastically reduce any impact of that form.

As for the implementation we make use of PyTorch [30], which provides a powerful platform for neural network models. Employing a gradient-based optimisation routine, we need to compute the partial derivatives of the cost function. This requires an application of the chain rule, which may not be trivial due to the matrix operations and numerical integration involved; to this end we rely upon PyTorch's support for automatic differentiation.

Additionally, a complementary routine for back propagation of derivatives through the QR-factorisation has been implemented based on [49], with details described in the supplementary material. Furthermore, we are using the L-BFGS-optimiser [42], modified to allow for a dynamically changing learning rate.

The procedure is summarised in Algorithm 1.

4 Experimental results

Here we illustrate the practical performance of the method, starting with a one-dimensional toy example and proceeding with more realistic CT examples.

4.1 One-dimensional toy example

To illustrate the method, we consider inference of the one-dimensional step function seen in Figure 3. The data set consist of 50 integrals computed over randomly chosen intervals in the domain $[0, 1]$, contaminated by Gaussian noise with standard deviation 0.001. For the latent mapping we are using a neural network with four layers and $(1, 5, 4, 1)$ neurons, employing the hyperbolic tangent activation function after the two hidden layers. Furthermore, we are using $N_t = 100$ uniformly spaced points in the pre-training (14). The squared exponential covariance function (8) is used in both the standard GP and the proposed method.

Figure 3 shows the result of the proposed method (red dashed-dotted) and the standard GP (blue dashed), together with their 95% credibility regions. Obviously, the standard GP suffers from its embedded smoothness assumptions and it is struggling with the step, which is reflected in the oscillations and the wide credibility region. It should be stressed that this problem differs notably as opposed to considering point measurements from a smooth function – with complexity added in both the measurement model and the function itself, it becomes significantly more challenging. Nevertheless, the proposed method performs clearly better than the standard GP, obtaining a good estimate of the true function.

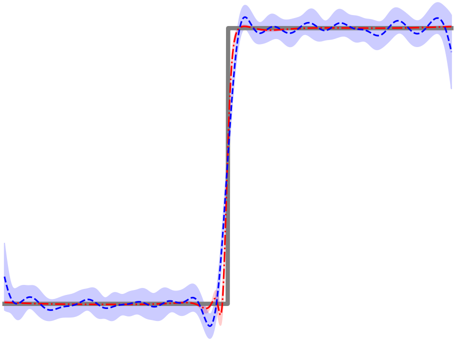


Figure 3: One-dimensional toy example of inferring a step function from 50 integral measurements over randomly chosen intervals in $[0, 1]$. True function in solid grey, standard GP in blue (dashed) and the proposed method in red (dash-dotted). The shades indicate the 95% credibility regions, which is much tighter for the proposed method.

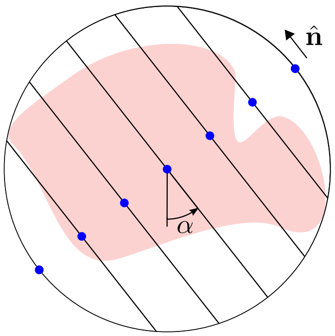


Figure 4: Measurement setup in X-ray computed tomography; illustration of a projection taken from the angle α, with the target object shaded. All lines in the projection share the same unit vector $\hat{\mathbf{n}}$, while the centre points are different (blue dots). The integration radius r_i is the distance on the line from the centre point $\mathbf{x}_i^0$ to the circle.

4.2 Computed tomography experiments

Here, we test the performance on two-dimensional CT problems. CT provides a good demonstration for our method as it involves line integral measurements of a quantity that can have discrete or sharp changes. We compare our proposed method against the filtered back projection (FBP) algorithm. For decades FBP has served as a state-of-the-art method, in no small part due to the fact that it outperforms iterative optimisation-based alternatives in terms of computation time. However, FBP is sensitive to noise and demonstrates a relatively poor performance for small data sets – also referred to as *limited data*. The limited data problem is interesting for several reasons, including: keeping the radiation doses small; efficient use of scanning devices; geometric setup restrictions (as in mammography).

In the practical scanning procedure, the data is collected as a set of *projections*, each of which defines a number of parallel lines sharing the same projection angle α. The projection width is determined by the maximum object width $w_{\max}$, so it is deduced that the entire object is located within a circle of radius $w_{\max}/2$. Exploiting this knowledge, the integration radii are found by identifying the intersections between the circle and the straight lines defined by the centre points and the unit vector; see Figure 4 for an illustration of the geometry.

In both the examples presented in this section, the neural networks have five layers with $(2, 30, 20, 6, 1)$ neurons and the hyperbolic tangent as activation function after the three hidden layers. Note that this structure is far from optimal and could most likely be improved with a more careful design. The input domain is normalised to $[-1, 1] \times [-1, 1]$, and the pre-training is using $N_t = 10^4$ uniformly spaced points. Also, all GPs are using the squared exponential covariance function (8). We compare the results to FBP reconstructions computed with the `iradon` command from the `skimage` module in Python [48]. The simulated data is generated with the corresponding `radon` command using a high-resolved version of the ground truth image.

Simulated data

As a simulated example we consider the Shepp-Logan phantom [**SheppLogan**]. The data consist of 9 projections evenly spaced in $[0, 160]^\circ$ with 185 lines each, yielding a total of 1 665 measurements. Furthermore, Gaussian noise with standard deviation 0.001 is added on top.

Figure 5 shows the ground truth image along with the reconstructions obtained with FBP and our proposed method, respectively. The drawback of the FPB in this case is obvious, as is seen from the distortions present both inside and outside the main ellipse. The GP model, on the other hand, is much more homogeneous within the respective regions. However, some blurriness is observed.

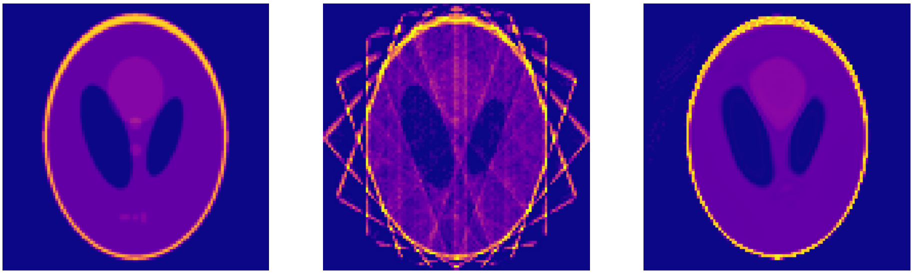


Figure 5: Left: Shepp-Logan phantom. Middle: FBP reconstruction. Right: proposed method.

Real CT data

As a real-world example, we consider the carved cheese data set provided by the Finish Inverse Problems Society, freely available and documented online [4, 11]. The data is down-sampled to contain 15 projections evenly spaced in $[0, 336]^\circ$ with 140 measurements each, hence 2 100 in total.

The result is seen in Figure 1; the leftmost plot shows a dense FBP reconstruction obtained from the complete set of 360 projections each with 2 240 measurements, that is more than 800 000 measurements in total. Hence, it is considered close to ground truth. Regarding the other reconstructions, the performance is similar to what we observed in the previous experiment. Clearly, the GP model produces a solution in which the different regions are better distinguished. There are some parts with blurry elements, primarily near the boundaries of the characters; it is likely that these effects could be overcome by a more well-designed network structure.

5 Related work

The use of input transformations in the covariance function is by no means a new construction; it is used in modelling solar radiation patterns [35] and to impose periodicity [22], which in turn is exploited in modelling of the atmospheric carbon dioxide concentration [33] and for long-term forecasting [15]. In [43], a linear input transformation is used for dimensionality reduction in sparse GPs. Another closely related approach is to transform the GP *outputs*, which relaxes the embedded Gaussianity assumptions [44].

Incorporation of deep learning into GPs has a long history as well. The neural network covariance function [28] is particularly notable, encoding a one-layer neural network with infinitely many neurons. Another area that has gained a lot of interest in recent years is constituted by the *deep GPs* [9, 10], where a series of GPs are combined in a network structure. However, the computational demand is rather intricate; scalable extensions of this model are developed in [8, 34], with variational inference being a key component.

As we have already mentioned, the foundation that this work relies upon is a technique referred to as manifold GPs [5] or deep kernel learning [54], both of which describe more or less the same procedure in slightly different contexts. An interesting extension is found in [53], where the framework is generalised to a broader class of problems using stochastic variational inference. Another similar approach considers a model customised for recurrent structures [40], where the performance using standard covariance functions is poor. In [14], a low-parameterised relative to deep kernel learning is developed with focus on convolutional neural networks. The construction has also gained interest as a potential tool in Bayesian optimisation [39, 55]. Closely related viewpoints are presented in [12, 26], which both consider the relation between GPs and wide deep neural networks.

The vast majority of GP models developed are concerned with point measurements; although integral measurements are not as common, they are present in relevant real-world applications, including CT reconstruction used for demonstration in this work. The CT problem has been successfully attacked from several different angles, using deep learning techniques [1, 2, 18, 31, 56] and statistical methods [3, 17, 25, 38] including the GP [32], but not previously with deep kernel learning. Another area of rising importance is strain field estimation based on the longitudinal ray transform [27, 36, 37], which constitute a line integral of the projected strain tensor. Since it involves the reconstruction of a multidimensional function, it is a technically more challenging problem than the CT equivalent. GPs tailored to satisfy the physical constraints of the strain field have been used to this end [19, 23], but so far no deep learning based techniques; the proposed method serves as an interesting extension. Yet another example of integral measurements in GPs are found within stochastic optimisation [20], where the secant condition used in quasi-Newton methods is replaced by its exact counterpart; this approach has shown promising results in nonlinear system identification [51].

6 Conclusion and future work

In this work we have presented a method that applies deep kernel learning to problems with integral measurements. We proposed utilising a basis function expansion to make the computations practically feasible, and pre-training of the neural network to improve the result of the joint parameter training.

The method was illustrated on both simulated and real data from X-ray computed tomography, indicating a promising potential.

Future work may focus on customisation of the neural network structures. Having paid a fairly limited attention to this important part of the model, we believe that the room for improvement is significant. Moreover, although the neural network is one possible choice of latent mapping, it is by no means the only one. Other alternatives are also worth exploring, as well as their potential combinations. As mentioned in Section 3.2 we did restrict ourselves to a single latent output to reduce the computational burden; however, a well-designed combination of several outputs with differing mappings is likely to improve the performance. Also, extensions of deep kernel learning aimed at reducing the risk of over-fitting should be explored to further improve the robustness.

7 Acknowledgements

This research was financially supported by the Swedish Foundation for Strategic Research (SSF) via the project *ASSEMBLE* (contract number: RIT15-0012).

8 Supplementary material

8.1 Selecting the domain size

Here we discuss the selection of the domain size of Ω, which is determined by the parameters L_k used in building the basis functions (11). A basic requirement is that L_k should be clearly larger than the maximum absolute size of $u_k(\mathbf{x})$ to avoid undesired impact of the Dirichlet conditions used in the eigenvalue problem (10). Apart from this, the size of L_k determines the quality of the approximation specified in the frequency domain; given a fixed number of basis function in direction k, increasing L_k yields a higher frequency resolution $\frac{\pi}{2L_k}$ in that direction, while at the same time it is reducing the frequency range $[\frac{\pi}{2L_k}, \frac{\tilde{m}\pi}{2L_k}]$. A reasonable approach is to select L_k with respect to the spectral frequency $S(\cdot)$, so that the domain covers the vast majority of the spectral "mass" (equivalently to how confidence regions covers different amounts of the probability mass). This is dependent on the lengthscale parameters, and we suggest selecting L_k such that

$$\alpha l_k^{-1} = \max_k c_{kj}, \tag{15}$$

where the parameter α is chosen with respect to the spectral density of the covariance function used. For instance, the value $\alpha = 5$ is reasonable for the squared exponential covariance function (8) and yields a coverage of more than 99.9%. Note that since L_k is not part of the optimised parameters, the recalculation modifies the definition of the cost function. However, that effect is negligible and the strategy has proven very useful in practise.

8.2 Numerical Implementation

The numerical robustness can be improved using the QR-factorisation, considering both the computations of the loss function and the predictions. For instance, the LOO-CV procedure requires $(\Phi\Lambda\Phi^\mathsf{T} + \sigma^2 I)^{-1}$ and $(\Phi\Lambda\Phi^\mathsf{T} + \sigma^2 I)^{-1}\mathbf{y}$. To that end, we first compute the matrix R in the QR-factorisation

$$QR = \begin{bmatrix} \Lambda^{1/2}\Phi^\mathsf{T} \\ \sigma I \end{bmatrix}. \tag{16}$$

Since Q is a unitary matrix, it follows that $R^\mathsf{T} R = \Phi^\mathsf{T}\Lambda\Phi + \sigma^2 I$ and so the desired quantities can be found using efficient forward and backward substitutions [16].

Since PyTorch's automatic differentiation is being used to provide the partial derivatives of the cost function with respect to the parameters θ, a 'backwards' method is required for the QR-factorisation. Although a QR algorithm is implemented in PyTorch, it does not have a backwards method in its current stable release. Given the partial derivative of the cost function C with respect to R, the backwards algorithm needs to compute the partial derivates of the cost function with respect to the elements of A, where $QR = A$. Algorithm 2 provides a backwards method that can be added to the QR function in PyTorch and is based upon the equations presented in [49].

Algorithm 2 Backwards Method for QR

```
Input: $\frac{\partial C}{\partial R}$, $Q$, and $R$
Output: $\frac{\partial C}{\partial A}$
Compute the psuedoinverse of $R$: $R^{+} = (R^\top R)^{-1} R^\top$
Compute $\beta = \left(R\frac{\partial C}{\partial R} - \frac{\partial C}{\partial R} R^\top\right)$
Extract the lower triangular matrix below the main diagonal: $\Gamma = \mathrm{tril}(\beta, -1)$
Compute the output: $\frac{\partial C}{\partial A} = Q\left(\frac{\partial C}{\partial R} + \Gamma R^{+\top}\right)$
```

References

[1] J. Adler and O. Öktem. "Learned Primal-Dual Reconstruction". In: *IEEE Transactions on Medical Imaging* 37.6 (2018), pp. 1322–1332.

[2] J. Adler and O. Öktem. "Solving ill-posed inverse problems using iterative deep neural networks". In: *Inverse Problems* 33.12 (2017), p. 124007.

[3] C. A. Bouman and K. Sauer. "A unified approach to statistical tomography using coordinate descent optimization". In: *IEEE Transactions on Image Processing* 5.3 (1996), pp. 480–492.

[4] T. A. Bubba, M. Juvonen, J. Lehtonen, M. März, A. Meaney, Z. Purisha and S. Siltanen. *Tomographic X-ray data of carved cheese.* Tech. rep. arXiv:1705.05732v2, June 2017.

[5] R. Calandra, J. Peters, C. E. Rasmussen and M. P. Deisenroth. "Manifold Gaussian Processes for Regression". In: *In Proceedings of the International Joint Conference on Neural Networks (IJCNN).* 2016, pp. 3338–3345.

[6] S. C. Chapra. *Applied Numerical Methods with MATLAB for engineers and scientists, 4th edition.* McGraw Hill, 2017, p. 720.

[7] A. M. Cormack. "Representation of a function by its line integrals, with some radiological applications". In: *Journal of Applied physics* 34.9 (1963), pp. 2722–2727.

[8] Z. Dai, A. Damianou, J. Gonzalez and N. D. Lawrence. "Variationally Auto-Encoded Deep Gaussian Processes". In: *Proceedings of the International Conference on Learning Representations (ICLR).* Vol. 3. 2016.

[9] A. C. Damianou. "Deep Gaussian processes and variational propagation of uncertainty". PhD thesis. Department of neuroscience, University of Sheffield, 2015.

[10] A. C. Damianou and N. D. Lawrence. "Deep Gaussian Processes". In: *International Conference on Artificial Intelligence and Statistics (AISTATS).* 2013.

[11] *Finish Inverse Problems Society.* `https://www.fips.fi/dataset.php`. Accessed: 2019-04-20.

[12] A. G. de G. Matthews, J. Hron, M. Rowland, R. E. Turner and Z. Ghahramani. "Gaussian Process Behaviour in Wide Deep Neural Networks". In: *International Conference on Learning Representations (ICLR).* 2018.

[13] R. Garnett. *Lecture 11: Bayesian Quadrature.* University Lecture. Accessed: 2019-05-01. Available from: http://www.cse.wustl.edu/~garnett/cse515t/spring_2018/files/lecture_notes/11.pdf. 2018.

[14] A. Garriga-Alonso, C. E. Rasmussen and L. Aitchison. "Deep Convolutional Networks as shallow Gaussian Processes". In: *International Conference on Learning Representations (ICLR).* 2019.

[15] N. H. Ghassemi and M. P. Deisenroth. "Analytic Long-Term Forecasting with Periodic Gaussian Processes". In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS).* 2014.

[16] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press; fourth edition edition, 2013, p. 784.

[17] H. Haario, A. Kallonen, M. Laine, E. Niemi, Z. Purisha and S. Siltanen. "Shape recovery for sparse-data tomography". In: *Mathematical Methods in the Applied Sciences* 40.18 (2017), pp. 6649–6669.

[18] K. Hammernik, T. Würfl, T. Pock and A. Maier. "A Deep Learning Architecture for Limited-Angle Computed Tomography Reconstruction". In: *Bildverarbeitung für die Medizin 2017.* Springer Berlin Heidelberg, 2017, pp. 92–97.

[19] J. N. Hendriks, A. W. T. Gregg, C. M. Wensrich, A. S. Tremsin, T. Shinohara, M. Meylan, E. H. Kisi, V. Luzin and O. Kirsten. "Bragg-edge elastic strain tomography for in situ systems from energy-resolved neutron transmission imaging". In: *Physical Review Materials* 1.5 (Oct. 2017), p. 053802.

[20] P. Hennig and M. Kiefel. "Quasi-Newton methods: A new direction". In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 843–865.

[21] G. T. Herman. "Image reconstruction from projections". In: *Topics in Applied Physics* 32 (1979).

[22] D. J. C. MacKay. "Introduction to Gaussian Processes". In: *NATO ASI Series F Computer and Systems Sciences* 168 (1998), pp. 133–166.

[23] C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. "Probabilistic modelling and reconstruction of strain". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

[24] C. Jidling, N. Wahlström, A. Wills and T. B. Schön. "Linearly constrained Gaussian processes". In: *Advances in Neural Information Processing Systems (NIPS).* Los Angeles, US, 2017, pp. 1215–1224.

[25] J. Kaipio and E. Somersalo. *Statistical and computational inverse problems*. Vol. 160. Springer Science & Business Media, 2006.

[26] J. Lee, J. Sohl-dickstein, J. Pennington, R. Novak, S. Schoenholz and Y. Bahri. “Deep Neural Networks as Gaussian Processes”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[27] W. R. B. Lionheart and P. J. Withers. “Diffraction tomography of strain”. In: *Inverse Problems* 31.4 (2015), p. 045005.

[28] R. M. Neal. *Bayesian Learning for Neural Networks*. Vol. 118. Lecture Notes in Statistics. Springer New York, 1996. ISBN: 9780387947242.

[29] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes*. New York: McGraw-Hill Education, 1991.

[30] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga and A. Lerer. “Automatic differentiation in PyTorch”. In: *The future of gradient-based machine learning software and techniques – workshop at the Neural Information Processing Systems Conference (NIPS)*. 2017.

[31] D. M. Pelt, K. J. Batenburg and J. A. Sethian. “Improving Tomographic Reconstruction from Limited Data Using Mixed-Scale Dense Convolutional Neural Networks”. In: *Journal of Imaging* 4.11 (2018). ISSN: 2313-433X.

[32] Z. Purisha, C. Jidling, N. Wahlström, T. Schön and S. Särkkä. “Probabilistic approach to limited-data computed tomography reconstruction”. In: *Inverse Problems* 35.10 (2019), p. 105004.

[33] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT press, Cambridge, MA, 2006.

[34] H. Salimbeni and M. Deisenroth. “Doubly Stochastic Variational Inference for Deep Gaussian Processes”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2017, pp. 4588–4599.

[35] P. D. Sampson and P. Guttorp. “Nonparametric Estimation of Nonstationary Spatial Covariance Structure”. In: *Journal of the American Statistical Association* 87.417 (1992), pp. 108–119.

[36] J. R. Santisteban, L. Edwards, M. E. Fitzpatrick, A. Steuwer, P. J. Withers, M. Daymond, M. W. Johnson, N. Rhodes and E. M. Schooneveld. “Strain imaging by Bragg edge neutron transmission”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 481.1 (2002), pp. 765–768.

[37] J. R. Santisteban, L. Edwards, M. E. Fizpatrick, A. Steuwer and P. J. Withers. "Engineering applications of Bragg-edge neutron transmission". In: *Applied Physics* 74.1 (2002), pp. 1433–1436.

[38] K. Sauer, J. Sachs and C. Klifa. "Bayesian estimation of 3-D objects from few radiographs". In: *IEEE Transactions on Nuclear Science* 41.5 (1994), pp. 1780–1790.

[39] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. De Freitas. "Taking the human out of the loop: A review of bayesian optimization". In: *Proceedings of the IEEE* 104.1 (2016), pp. 148–175.

[40] M. Al-Shedivat, A. G. Wilson, Y. Saatchi, Z. Hu and E. P. Xing. "Learning Scalable Deep Kernels with Recurrent Structure". In: *Journal of Machine Learning Research* 18.82 (2017), pp. 1–37.

[41] L. A. Shepp and J. B. Kruskal. "Computerized tomography: the new medical X-ray technology". In: *American Mathematical Monthly* (1978), pp. 420–439.

[42] H.-J. M. Shi and D. Mudiger. *PyTorch-LBFGS: A PyTorch implementation of L-BFGS*. https://github.com/hjmshi/PyTorch-LBFGS. 2017.

[43] E. Snelson and Z. Ghahramani. "Variable noise and dimensionality reduction for sparse Gaussian processes". In: *Proceedings of Uncertainty in Artificial Intelligence (UAI)*. 2006.

[44] E. Snelson, Z. Ghahramani and C. E. Rasmussen. "Warped Gaussian Processes". In: *Advances in Neural Information Processing Systems (NIPS)*. 2004, pp. 337–344.

[45] A. Solin and M. Kok. "Know Your Boundaries: Constraining Gaussian Processes by Variational Harmonic Features". In: *Proceedings of Machine Learning Research*. Vol. 89. 2019, pp. 2193–2202.

[46] A. Solin and S. Särkkä. "Hilbert space methods for reduced-rank Gaussian process regression". In: *Statistics and Computing* (Aug. 2019).

[47] M. L. Stein. *Interpolation of Spatial Data*. Springer-Verlag New York, Feb. 1999, p. 249.

[48] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu and the scikit-image contributors. "scikit-image: image processing in Python". In: *PeerJ* 2 (2014), e453. ISSN: 2167-8359.

[49] S. Walter. "Structured higher-order algorithmic differentiation in the forward and reverse mode with application in optimum experimental design". PhD thesis. Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät II, 2012.

[50] C. M. Wensrich, J. N. Hendriks, A. Gregg, M. H. Meylan, V. Luzin and A. S. Tremsin. “Bragg-edge neutron transmission strain tomography for in situ loadings”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 383 (2016), pp. 52–58.

[51] A. G. Wills and T. B. Schön. “On the construction of probabilistic Newton-type algorithms”. In: *IEEE 56th Annual Conference on Decision and Control (CDC).* Melbourne, Australia, Dec. 2017, pp. 6499–6504.

[52] A. G. Wilson and R. Adams. “Gaussian process kernels for pattern discovery and extrapolation”. In: *International Conference on Machine Learning (ICML).* 2013, pp. 1067–1075.

[53] A. G. Wilson, Z. Hu, R. R. Salakhutdinov and E. P. Xing. “Stochastic variational deep kernel learning”. In: *Advances in Neural Information Processing Systems (NIPS).* 2016, pp. 2586–2594.

[54] A. G. Wilson, Z. Hu, R. Salakhutdinov and E. P. Xing. “Deep Kernel Learning”. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS).* Cadiz, Spain, May 2016, pp. 370–378.

[55] J. Wu, M. Poloczek, A. G. Wilson and P. Frazier. “Bayesian Optimization with Gradients”. In: *Advances in Neural Information Processing Systems (NIPS).* 2017, pp. 5267–5278.

[56] T. Würfl, F. C. Ghesu, V. Christlein and A. Maier. “Deep Learning Computed Tomography”. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI).* 2016, pp. 432–440. ISBN: 978-3-319-46726-9.

Paper V

Title

Neutron transmission strain tomography for non-constant strain-free lattice spacing.

Authors

Johannes Hendriks, Carl Jidling, Thomas B. Schön, Adrian Wills, Christopher Wensrich, and Erich Kisi.

Statement of contribution

The conceptual method to solve this problem was developed by Johannes Hendriks, Carl Jidling, Thomas Schön, and Adrian Wills, with the implementation undertaken by Johannes Hendriks. The technical background dealing with crystal structure and Bragg-edge neutron transmission techniques was provided by Chris Wensrich, Erich Kisi, and Johannes Hendriks. Overall, contributions to the work are well reflected in the author ordering. Carl Jidling provided essential background in nonlinear methods for Gaussian processes and techniques for hyper parameter optimisation.

Edited version of

J. N. Hendriks, C. Jidling, T. B. Schön, A. Wills, C. M. Wensrich and E. H. Kisi. “Neutron transmission strain tomography for non-constant stress-free lattice spacing”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 456 (2019), pp. 64–73.

Neutron transmission strain tomography for non-constant strain-free lattice spacing

Abstract

Recently, several algorithms for strain tomography from energy-resolved neutron transmission measurements have been proposed. These methods assume that the strain-free lattice spacing d_0 is a known constant limiting their application to the study of stresses generated by manufacturing and loading methods that do not alter this parameter. In this paper, we consider the more general problem of jointly reconstructing the strain and d_0 fields. A method for solving this inherently non-linear problem is presented that ensures the estimated strain field satisfies equilibrium and can include knowledge of boundary conditions. This method is tested on a simulated data set with realistic noise levels, demonstrating that it is possible to jointly reconstruct d_0 and the strain field.

1 Introduction

Energy-resolved neutron transmission methods can generate lower dimensional (one- or two-dimensional) images of strain from a higher dimensional (two- or three-dimensional) strain field within a polycrystalline material. The 'tomographic' reconstruction of an unknown strain field from these images can be used to study the residual strain and stress within engineering components. Residual stresses are those which remain after applied loads are removed (e.g. due to heat treatment, plastic deformation, etc.), and may have significant and unintended impact on a component's effective strength and service life — in particular its fatigue life. Measuring and quantifying these strains is important for the validation of predictive design tools, such as Finite Element Analysis, and to aid the development of novel manufacturing techniques — i.e. additive manufacturing.

These strain images are generated by analysing features known as Bragg-edges in the relative transmission of a neutron pulse through a sample. Bragg-edges are sudden increases in the intensity as a function of wavelength and occur when the scattering angle 2ϑ reaches 180°, beyond which no further coherent scattering can occur. The wavelength λ at which these Bragg-edges occur can be related to the lattice spacing d within the sample through Bragg's law: $\lambda = 2d \sin \vartheta$. Assuming minimal texture, this can be used to provide a relative measure of strain;

$$\langle \epsilon \rangle = \frac{d - d_0}{d_0}, \tag{1}$$

where d_0 is the strain-free lattice spacing and $\langle \epsilon \rangle$ is a through thickness average of the normal, elastic strain in the direction of the beam.

The determination of d_0 is a problem inherent to diffraction and transmission strain analysis. For specific cases where the loading mechanism does not result in changes to the strain-free lattice parameter, its value can be measured prior to loading and in the simplest case (e.g. for an annealed sample) a constant value throughout the sample can be assumed. Several algorithms for strain tomography assuming a known, constant strain-free lattice spacing have been developed. Reconstruction of axisymmetric strain fields is considered in [1, 2, 7, 17] and more general two-dimensional strain fields in [8, 10, 14].

Many manufacturing techniques (e.g. welding and additive manufacturing) can alter the lattice spacing; for example, as a result of inhomogeneously distributed phase changes (such as the Martensite transformation), or due to gradients in composition as a result of differing chemical states in the starting materials. Since the lattice spacing (in this case d_0) is sensitive to crystal structure and composition changes, the strain-free lattice parameter may vary throughout the sample. Ignoring variations in d_0 would cause severe degradation in the quality of a reconstructed strain field. In such cases, measuring d_0 is more challenging and has been achieved in neutron diffraction measurements by measuring additional directions of strain [6, 19] and by destructive methods where the strain is relieved by wire cutting the sample into a grid allowing the strain-free lattice spacing to be measured throughout the sample [24]. Although the latter of these two options could be applied to strain tomography it requires the destruction of the sample and creates an additional tomography problem, requiring another set of measurements to be acquired.

Here, we present a method capable of jointly reconstructing the strain field and the d_0 field from a single set of neutron transmission images. To achieve this both the strain and d_0 are modelled by a Gaussian process (see for example [25]) and equilibrium and boundary conditions are built into the strain model [15]. This extends the Gaussian process approach presented by [10, 14] to handle the inherently non-linear nature of this problem. A numerically tractable algorithm based on variational inference (see for example [5, 16]) is provided and the method is validated on a simulated data set.

2 Problem Statement

This paper focuses on the joint reconstruction of the strain field $\boldsymbol{\epsilon}(\mathbf{x})$ and a non-constant strain-free lattice parameter $d_0(\mathbf{x})$ from a set of neutron transmission images. Restricting the problem to two dimensions, gives the strain field as the symmetric tensor

$$\boldsymbol{\epsilon}(\mathbf{x}) = \begin{bmatrix} \epsilon_{xx}(\mathbf{x}) & \epsilon_{xy}(\mathbf{x}) \\ \epsilon_{xy}(\mathbf{x}) & \epsilon_{yy}(\mathbf{x}) \end{bmatrix}, \tag{2}$$

where $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^{\mathsf{T}}$. For brevity, the unique components of strain will be written as $\bar{\epsilon} = \begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} & \epsilon_{yy} \end{bmatrix}^{\mathsf{T}}$ with the coordinate $\mathbf{x}$ omitted where appropriate.

Here, we consider the lattice spacings d as the measurements rather than the standard approach which considers the relative strain of the form form (1). This allows the measurements to be explicitly related to both the strain and the strain-free lattice parameter through the Longitudinal Ray Transform (LRT) [18]:

$$y(\boldsymbol{\eta}) = d(\boldsymbol{\eta}) + e = \frac{1}{L}\int_0^L \bar{\mathbf{n}}\bar{\boldsymbol{\epsilon}}(\mathbf{p} + \hat{\mathbf{n}}s)d_0(\mathbf{p} + \hat{\mathbf{n}}s) + d_0(\mathbf{p} + \hat{\mathbf{n}}s)\,\mathrm{d}s + e. \tag{3}$$

where $e \sim \mathcal{N}(0, \sigma_n^2)$ and the geometry of each measurement is given by the parameter set $\boldsymbol{\eta} = \{\hat{\mathbf{n}}, L, \mathbf{p}\}$; with $\hat{\mathbf{n}}$ as the beam direction, L as the irradiation length, $\mathbf{p} = \begin{bmatrix} x_0 & y_0 \end{bmatrix}^{\mathsf{T}}$ as the point of initial intersection between the ray and the sample, and $\bar{\mathbf{n}} = \begin{bmatrix} \hat{\mathbf{n}}_1^2 & 2\hat{\mathbf{n}}_1\hat{\mathbf{n}}_2 & \hat{\mathbf{n}}_2^2 \end{bmatrix}$. See Figure 1 for the measurement geometry. These measurements are a non-linear function of the two phenomena we wish to estimate; $\boldsymbol{\epsilon}$ and d_0.

For details on the analysis of neutron transmission data to determine these lattice spacings the reader is referred to [27, 28, 31, 32]. It is also worth noting that the standard deviation σ_n of these measurements is available.

Furthermore, the strain field inside a sample is a physical property and as such it is subject to equilibrium and boundary conditions [26]. Therefore, it is natural to constrain estimates of the strain field to satisfy these conditions. Using Hooke's law the equilibrium conditions can be written directly in terms of strain. In two dimensions, this relies on an assumption of plane strain or plane stress. Plane stress is assumed for the remainder of this work, giving the equilibrium conditions as

$$\begin{aligned} \frac{\partial}{\partial x}(\epsilon_{xx} + \nu\epsilon_{yy}) + \frac{\partial}{\partial y}(1-\nu)\epsilon_{xy} = 0, \\ \frac{\partial}{\partial x}(\epsilon_{yy} + \nu\epsilon_{xx}) + \frac{\partial}{\partial y}(1-\nu)\epsilon_{xy} = 0, \end{aligned} \tag{4}$$

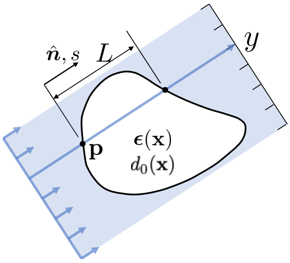


Figure 1: LRT measurement geometry. Each measurement made by a detector pixel is associated with a ray of direction $\hat{\mathbf{n}}$ that enters the sample at $\mathbf{p}$ and has a total irradiated length of L.

where ν is Poisson's ratio.

Boundary conditions, in particular the load free surfaces, may also be known. For an unloaded surface, the distribution of forces known as tractions will be zero. Through equilibrium this places additional linear constraints on the strain field, which, assuming plane stress, can be written as

$$\mathbf{0} = \begin{bmatrix} n_{\perp 1} & n_{\perp 2} & 0 \\ 0 & n_{\perp 1} & n_{\perp 2} \end{bmatrix} \begin{bmatrix} 1 & 0 & -\nu \\ 0 & 1+\nu & 0 \\ -\nu & 0 & 1 \end{bmatrix} \bar{\boldsymbol{\epsilon}}(\mathbf{x}_b), \tag{5}$$

where $\mathbf{x}_b$ is a point on an unloaded surface and $\mathbf{n}_\perp$ is the normal to the surface at this point.

An approach to enforcing equilibrium in the estimated strain field is to define a Gaussian process for the Airys stress function from which strain can be derived [14]. This non-parametric approach was demonstrated experimentally by [14] and compared to other parametric approaches by [10] with promising results. Boundary conditions in the form of (5) can be included in the estimation process as artificial measurements of zero traction [10].

We wish to extend this approach so that both the strain field and the strain-free lattice spacing can be estimated. As the measurements are a non-linear function of the unknowns we cannot directly apply the standard Gaussian process regression methods [25]. There exists several approaches to approximate Gaussian processes for non-linear functions; the Laplace approximation [4, 25], GP variational inference [30], and Markov Chain Monte Carlo methods (such as Elliptical Slice Sampling [21]). For these methods, the measurements are modelled as non-linear functions of the GP sampled at the measurement locations (known as latent function values). The latent function values that best[1] match the data are determined by one

[1] For a given criterion of best fit, whether it be marginal log likelihood, cross-validation, etc.

of the above methods. Then, Gaussian process regression is applied with the latent function values taking the place of measurements to determine the function values at the new locations of interest.

The non-static nature of the integral measurement model (3) makes it unclear how to express the measurements as a function of a finite set of latent function values, and hence the above approaches to approximating the GP for non-linear measurements cannot be applied directly. In the following section, we utilise an finite basis function approximation to the GP, and by viewing the problem from an alternate perspective we show how variational inference can be used to solve this non-linear problem.

3 Method

The method presented here is to define a Gaussian process model for the strain field and the strain-free lattice spacing. This Gaussian process model is then approximated using a Hilbert space approximation [14, 29]. This has two benefits; firstly it removes the need for numerical integration of the covariance function (as discussed by [14, 15]), and secondly it allows us to reformulate the problem as a set of basis functions with unknown coefficients. Variational inference can then be used to learn the coefficients from the LRT measurements and artificial measurements of zero traction.

3.1 Gaussian Process model

The Gaussian process (GP) is a Gaussian distribution of spatially correlated functions;

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right). \tag{6}$$

The characteristics of the functions belonging to this distribution are governed by a mean function $m(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$. The covariance function describes the correlation between the function values $f(\mathbf{x})$ and $f(\mathbf{x}')$ at any two points $\mathbf{x}$ and $\mathbf{x}'$. Careful design of the covariance function can ensure that only functions satisfying desired characteristics belong to the distribution.

Here, we wish to design the covariance function to ensure that only strain fields satisfying equilibrium are contained in the GP prior. Following the formulation in [14] a GP model for the Airys stress functions is defined; $\varphi(\mathbf{x}) \sim \mathcal{GP}\left(0, k_\varphi(\mathbf{x}, \mathbf{x}')\right)$. Under the assumption that the sample is plane stress, isotropic, and contiguous, the Airy's stress functions can be related to strain through the mapping

$$\bar{\boldsymbol{\epsilon}}(\mathbf{x}) = \mathcal{V}^{\mathbf{x}}\varphi(\mathbf{x}), \qquad \mathcal{V}^{\mathbf{x}} = \begin{bmatrix} \frac{\partial^2}{\partial y^2} - \nu\frac{\partial^2}{\partial x^2} \\ -(1+\nu)\frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial x^2} - \nu\frac{\partial^2}{\partial y^2} \end{bmatrix}, \tag{7}$$

where $\mathcal{V}$ is a linear operator, and the superscript denotes which variable the operator acts on. As GPs are closed under linear operators [13, 14, 23, 33] a GP model for strain that satisfies equilibrium can now be defined;

$$\bar{\boldsymbol{\epsilon}}(\mathbf{x}) \sim \mathcal{GP}\left(0, \mathcal{V}^{\mathbf{x}} k_{\varphi}(\mathbf{x}, \mathbf{x}')\mathcal{V}^{\mathbf{x}'\mathsf{T}}\right), \tag{8}$$

where a prior mean function of zero has been chosen.

Additionally, d_0 function is also modelled by a GP;

$$d_0(\mathbf{x}) \sim \mathcal{GP}(\bar{d}_0, k_{d_0}(\mathbf{x}, \mathbf{x}')). \tag{9}$$

where the prior mean $\bar{d}_0$ is chosen to be close to the expected theoretical strain-free lattice spacing for the material used or a measured average in a strain-free sample. The choice of prior mean function does not mean that we believe the d_0 and $\bar{\boldsymbol{\epsilon}}$ functions to be a particular value, but rather that we do not have any information to suggest otherwise. After the inclusion of measurement information, the mean of the posterior estimate will be updated.

There exists a number of options for the base covariance functions $k_{\varphi}(\mathbf{x}, \mathbf{x}')$ and $k_{d_0}(\mathbf{x}, \mathbf{x}')$, with both the squared-exponential and the Matérn covariance functions having been successfully used for strain estimation [10, 14]. For a more thorough discourse on available covariance functions the reader is referred to [25].

Having defined suitable GP models for the strain and d_0 fields we now wish to estimate these fields from the LRT and traction measurements. However, the LRT is a non-linear function of these fields and consequently a closed form solution does not exist. The following presents a method for obtaining these estimates that approximates the GP by a finite number of basis functions allowing variational inference to be applied.

3.2 Hilbert Space Approximation to the GP Prior

Here, we make use of the approximation method proposed by [29] and demonstrated to be suitable for the problem of strain tomography [14]. This method approximates our covariance function by a finite sum of m basis functions;

$$k(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{m} \phi_i(\mathbf{x}) S(\boldsymbol{\lambda}_j) \phi_j(\mathbf{x}'), \tag{10}$$

where S is the spectral density of the covariance function. For a stationary covariance function $k = k(\mathbf{r})$, where $\mathbf{r} = \mathbf{x} - \mathbf{x}'$, the spectral density and the basis functions are given by;

$$S(\boldsymbol{\omega}) = \int k(\mathbf{r}) e^{-i\boldsymbol{\omega}^\mathsf{T}\mathbf{r}}\,\mathrm{d}\mathbf{r}, \quad \phi_j = \frac{1}{\sqrt{L_x L_y}} \sin(\lambda_{x,j}(x + L_x)) \sin(\lambda_{y,j}(y + L_y)), \tag{11}$$

where L_x and L_y control the domain size, and $\boldsymbol{\lambda} = [\lambda_x, \lambda_y]^\mathsf{T}$ encodes spatial frequencies of the basis functions. The basis functions are chosen as a solution to the Dirichlet boundary conditions on a rectangular domain, which is a natural choice for the Laplace eigenvalue problem that needs to be solved to approximate the GP [14]. The parameters $\theta = \{l_x, l_y, \sigma_f\}$ are commonly called 'hyperparameters' and can be chosen by optimisation (as discussed in Section 5.2). For our application the domain size and spatial frequencies are chosen such that the basis functions spanned a region where their spectral densities, were greater than a minimum threshold. This helps to ensure that the dominant frequencies of the response are captured while maintaining numerical stability.

At this stage, the alternative view point of Bayesian linear regression can be taken. This approach models the unknown function by a set of basis functions with Gaussian coefficients;

$$f(\mathbf{x}) = \sum_{j=1}^{m} \phi_j(\mathbf{x}) w_j = \boldsymbol{\phi}(\mathbf{x})\mathbf{w}, \qquad w_j \sim \mathcal{N}(\mu_j, S(\boldsymbol{\lambda}_j)), \tag{12}$$

where $\boldsymbol{\phi}(\mathbf{x})$ and $\mathbf{w}$ have dimensions $[1, m]$ and $[m, 1]$, respectively. This gives the following model for the strain field $\bar{\boldsymbol{\epsilon}}(\mathbf{x})$ and the strain-free lattice spacing $d_0(\mathbf{x})$;

$$\begin{aligned}
&\bar{\boldsymbol{\epsilon}}_*(\mathbf{x}) = \boldsymbol{\phi}_\epsilon \mathbf{w}_\varphi, \\
&\phi_{\epsilon,j}(\mathbf{x}) = \mathcal{V}^{\mathbf{x}} \phi_{\varphi,j}(\mathbf{x}), \\
&\phi_{\varphi,j} = \frac{1}{\sqrt{L_{\varphi x} L_{\varphi y}}} \sin(\lambda_{\varphi x,j}(x + L_{\varphi x})) \sin(\lambda_{\varphi y,j}(y + L_{\varphi y})), \\
&d_{0*}(\mathbf{x}) = \boldsymbol{\phi}_{d_0} \mathbf{w}_{d_0}, \\
&\phi_{d_0,k}(\mathbf{x}) = \frac{1}{\sqrt{L_{d_0 x} L_{d_0 y}}} \sin(\lambda_{d_0 x,k}(x + L_{d_0 x})) \sin(\lambda_{d_0 y,k}(y + L_{d_0 y})),
\end{aligned} \tag{13}$$

where the unknown coefficients are independently normally distributed to approximate our GP model:

$$w_{\varphi,j} \sim \mathcal{N}(0, S_\varphi(\boldsymbol{\lambda}_{\varphi,j})) \qquad \text{and} \qquad w_{d_0,j} \sim \mathcal{N}(\mu_{d_0,j}, S_{d_0}(\boldsymbol{\lambda}_{d_0,j})).$$

Where the means $\mu_{d_0,j}$ are chosen so that the prior has the constant value $\bar{d}_0$. In this work, basis functions and parameters corresponding to the d_0 field will be denoted by the subscript d_0 and the subscript k will be used as an index. Likewise, basis functions and parameters corresponding to the Airys stress function will be denoted by the subscript d_0 and the subscript j will be used as an index. The expanded expressions for $\boldsymbol{\phi}_\epsilon$ are given in Appendix A.

Using the LRT (3) we can write a model for a predicted measurement as a non-linear function of the unknown coefficients;

$$
\begin{aligned}
y_* = \frac{1}{L}\int_0^L \bar{\mathbf{n}} & \left(\sum_j \sum_k \phi_{\epsilon,j}(\mathbf{p}+\hat{\mathbf{n}}s) w_{\varphi,j} \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s) w_{d_0,k} \right) \\
& + \left(\sum_k \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s) w_{d_0,k} \right) \mathrm{d}s \\
= g_y(\mathbf{w}_\varphi, \mathbf{w}_{d_0}, \boldsymbol{\eta}), &
\end{aligned}
\tag{14}
$$

where we have restricted ourselves to a single measurement to simplify the notation. These integrals can be analytically evaluated and the equations are given in Appendix A. Predictions of the boundary tractions $\mathbf{y}_t$ at a boundary location $\mathbf{x}_b$ with surface normal $\mathbf{n}_\perp$ can be written as a linear function of the unknown coefficients;

$$
\begin{aligned}
y_{t*} &= \underbrace{\begin{bmatrix} n_{\perp 1} & n_{\perp 2} & 0 \\ 0 & n_{\perp 1} & n_{\perp 2} \end{bmatrix}}_{\mathbf{T}} \boldsymbol{\phi}_\varphi(\mathbf{x})\mathbf{w}_\varphi = \mathbf{T}(\mathbf{n}_\perp)\boldsymbol{\phi}_\varphi(\mathbf{x}_b)\mathbf{w}_\varphi \\
&= g_t(\mathbf{w}, \mathbf{x}_b, \mathbf{n}_\perp).
\end{aligned}
\tag{15}
$$

The coefficients $\mathbf{w}_\varphi$ and $\mathbf{w}_{d_0}$ are random variables; as such the predictions $\bar{\boldsymbol{\epsilon}}_*$, y_*, and y_{t*} are also random variables. The problem now is to determine the distribution of the coefficients given a set of LRT and traction measurements. This problem is now in a form allowing variational inference to be used to approximate a solution to the non-linear problem.

3.3 Variational Inference

Variational inference [5, 16] provides an approximation to the posterior distribution by assuming that it has a certain functional form that contain unknown parameters. These unknown parameters are found using optimization, where some distance measure is minimized. We will in this section provide the details enabling the use of variational inference in solving our problem.

Given n transmission measurements and n_t traction measurements, such that a vector of all measurements is given by $\mathbf{Y} = [y_1, \ldots, y_n, y_{t,1}, \ldots, y_{t,n_t}]^\mathsf{T}$, the problem can be written as having prior and likelihood

$$p(\mathbf{w}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}_p) \ \text{ and } \ p(\mathbf{Y}|\mathbf{w}) \sim \mathcal{N}(\mathbf{Y}|g(\mathbf{w}), \boldsymbol{\Sigma}_n), \tag{16}$$

where $\mathbf{w} = \begin{bmatrix}\mathbf{w}_{d_0}^{\mathsf{T}} & \mathbf{w}_{\varphi}^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}}$, $\boldsymbol{\mu}$ is a vector of all the prior means, and $\boldsymbol{\Sigma}_p$ is a matrix with the coefficients prior variance on the diagonals. Here, $g(\cdot)$ is the combined measurement model that expresses the measurement vector $\mathbf{Y}$ as a function of the coefficients. This function is constructed using both (14) and (15). Finally, $\boldsymbol{\Sigma}_n = \text{diag}(\sigma_n^2 I_{n\times n}, \sigma_t^2 I_{n_t\times n_t})$, where σ_t^2 is a small variance placed on the artificial traction measurements added for numerical reasons.

The non-linear measurement function $g(\cdot)$ makes the likelihood intractable as the prior and likelihood are no longer conjugate. Consequently, the posterior $p(\mathbf{w}|\mathbf{Y})$ is also intractable and so we find an approximate solution using variational inference [16]. The idea is to approximate the true posterior by the Gaussian distribution $q(\mathbf{w}) \sim \mathcal{N}(\hat{\mathbf{w}}, \mathbf{C})$, and find the mean $\hat{\mathbf{w}}$ and covariance $\mathbf{C}$ for this distribution that maximise the Free Energy $\mathcal{F}$. The Free Energy places a lower bound on the log marginal likelihood and hence provides a measure of how well our posterior fits the data;

$$\log p(\mathbf{Y}) \geq \mathbb{E}\left[\log p(\mathbf{Y}|\mathbf{W})\right] - KL\left[q(\mathbf{w})||p(\mathbf{w}|\mathbf{Y})\right] = \mathcal{F}, \tag{17}$$

where, in this case, $\mathbb{E}[\cdot]$ is the expected value with respect to the approximate posterior $q(\mathbf{w})$ and $KL[\cdot]$ is the Kullback Leibler divergence which provides a measure of difference between the approximate posterior and the true posterior. These terms can be evaluated as [30];

$$\begin{aligned}
\mathbb{E}\big[\log p(\mathbf{Y}|\mathbf{W})\big] &= \frac{1}{2}\big[N\log 2\pi + \log|\boldsymbol{\Sigma}_n| \\
&\quad + (\mathbf{Y} - \mathbb{E}\left[g(\mathbf{w})\right])^{\mathsf{T}}\boldsymbol{\Sigma}_n^{-1}(\mathbf{Y} - \mathbb{E}\left[g(\mathbf{w})\right])\big], \\
KL\big[q(\mathbf{w})||p(\mathbf{w}|\mathbf{Y})\big] &= \frac{1}{2}\big[\text{tr}(\boldsymbol{\Sigma}_p^{-1}\mathbf{C}) + \big(\boldsymbol{\mu} - \hat{\mathbf{w}}\big)^{\mathsf{T}}\boldsymbol{\Sigma}_p^{-1}\big(\boldsymbol{\mu} - \hat{\mathbf{w}}\big) \\
&\quad - \log|\mathbf{C}| + \log|\boldsymbol{\Sigma}_p| - N\big],
\end{aligned} \tag{18}$$

where $N = n + n_t$. Here, the expectation of the non-linear function $\mathbb{E}\left[g(\mathbf{w})\right]$ is intractable [30] and so the expected maximum is used $\hat{\mathbf{Y}} = g(\hat{\mathbf{w}})$;

$$\begin{aligned}
\mathcal{F} \approx -\frac{1}{2}\big[N\log 2\pi + \log|\boldsymbol{\Sigma}_n| - \log|\mathbf{C}| + \log|\boldsymbol{\Sigma}_p| + (\mathbf{Y} - g(\hat{\mathbf{w}}))^{\mathsf{T}}\boldsymbol{\Sigma}_n^{-1}(\mathbf{Y} - g(\hat{\mathbf{w}})) \\
+ \left(\boldsymbol{\mu} - \hat{\mathbf{w}}\right)^{\mathsf{T}}\boldsymbol{\Sigma}_p^{-1}\left(\boldsymbol{\mu} - \hat{\mathbf{w}}\right)\big].
\end{aligned} \tag{19}$$

The optimal posterior mean is chosen to maximise the Free Energy. To perform this optimisation a modified Newton's method is used where the step direction is $\mathbf{q} = -\mathbf{H}^{-1}\mathbf{g}$ and we can calculate the gradient, $\mathbf{g}$, and Hessian, $\mathbf{H}$, of the cost as

$$\begin{aligned} \mathbf{g} &= \mathbf{J}^{\mathsf{T}}\boldsymbol{\Sigma}_n^{-1}(\mathbf{Y} - g(\hat{\mathbf{w}})) - \boldsymbol{\Sigma}_p^{-1}\hat{\mathbf{w}}, \\ \mathbf{H} &= -\mathbf{J}^{\mathsf{T}}\boldsymbol{\Sigma}_n^{-1}\mathbf{J} + \frac{\partial \mathbf{J}^{\mathsf{T}}}{\partial \mathbf{w}}\Sigma_n^{-1}(\mathbf{Y} - g(\hat{\mathbf{w}})) - \boldsymbol{\Sigma}_p^{-1}, \end{aligned} \tag{20}$$

where $\mathbf{J} = \begin{bmatrix} \frac{\partial \hat{\mathbf{Y}}}{\partial \hat{\mathbf{w}}}^{\mathsf{T}} & \frac{\partial \hat{\mathbf{Y}}_{\mathbf{t}}}{\partial \hat{\mathbf{w}}}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}}$, and the derivatives and second derivatives are given in Appendix B. At each iteration we update the coefficients according to

$$\hat{\mathbf{w}}_{k+1} = (1-\alpha)\hat{\mathbf{w}}_k + \alpha\mathbf{q} + \alpha\boldsymbol{\mu}, \tag{21}$$

A backwards line search is used to ensure that $\mathcal{F}$ is increased in each iteration. Once the optimal posterior mean is found, the covariance can be found by setting $\frac{\partial \mathcal{F}}{\partial \mathbf{C}} = 0$ and linearising about $\hat{\mathbf{w}}$ [30], giving;

$$\mathbf{C} = \left[\boldsymbol{\Sigma}_p^{-1} + \mathbf{J}^{\mathsf{T}}\Sigma_n^{-1}\mathbf{J}\right]^{-1}. \tag{22}$$

Pseudo-code for an algorithm to find approximate distribution of the coefficients $q(\mathbf{w}) \sim \mathcal{N}(\hat{\mathbf{w}}, \mathbf{C})$ is given in Algorithm 1. Once the coefficients are found, estimates of the strain and d_0 fields can be estimated. The approximate poster mean and variance for the strain field and strain-free lattice spacing can be computed as

$$\begin{aligned} \begin{bmatrix} \hat{\mathbf{d}}_0(\mathbf{x}) \\ \hat{\boldsymbol{\epsilon}} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{\phi}_\epsilon(\mathbf{x}) & 0 \\ 0 & \boldsymbol{\phi}_\epsilon(\mathbf{x}) \end{bmatrix} \hat{\mathbf{w}}, \\ \hat{\boldsymbol{\Sigma}} &= \begin{bmatrix} \boldsymbol{\phi}_\epsilon(\mathbf{x}) & 0 \\ 0 & \boldsymbol{\phi}_\epsilon(\mathbf{x}) \end{bmatrix} \mathbf{C} \begin{bmatrix} \boldsymbol{\phi}_\epsilon(\mathbf{x}) & 0 \\ 0 & \boldsymbol{\phi}_\epsilon(\mathbf{x}) \end{bmatrix}^{\mathsf{T}}, \end{aligned} \tag{23}$$

where $\hat{\boldsymbol{\Sigma}}$ is the joint covariance of the strain and d_0 estimates. Next, this method is demonstrated on a set of measurements simulated from a theoretical cantilever beam strain field and an artificial d_0 field.

4 Simulation Results

The method's ability to jointly reconstruct the strain field and a d_0 field is demonstrated using simulated measurements. Reconstructions from measurements simulated through two strain fields is shown; the Saint-Venant approximate strain field for a cantilver beam, and a Finite Element Analysis (FEA) strain field from an in-situ loaded C-shape. Additionally, the consequences of ignoring the d_0 variation on the reconstruction are shown by using the linear measurement model and Gaussian process regression method presented by [14] with the addition of traction constraints as shown in [10]. Matlab code to run both examples can be found on Github [9].

Algorithm 1 Variational inference algorithm for finding the coefficients $q(\mathbf{w}) \sim \mathcal{N}(\hat{\mathbf{w}}, \mathbf{C})$. Requires the hyperparameters θ, the specified number of basis functions m_φ and m_{d_0}, the LRT measurement information $\{y_i, \boldsymbol{\eta}_i | \forall i = 1, \ldots, n\}$ and the boundary traction information $\{y_{t,i} = 0, \mathbf{x}_{b,i}, \mathbf{n}_{\perp,i} | \forall i = 1, \ldots, n_t\}$.

```
procedure Find Coefficients
    Compute the basis functions for the LRT measurements using Equation (14)
    Compute the basis functions for the traction measurements using Equation (15)
    Build prior variance $\boldsymbol{\Sigma}_p$
    Initialise the coefficients $\hat{\mathbf{w}}_1$
    set $k = 1$
    while Stopping criteria not met do
        Compute the gradient $\mathbf{g}$ and Hessian $\mathbf{H}$ linearised about $\hat{\mathbf{w}}_k$ according to Equation (20)
        Calculate $\hat{\mathbf{w}}_{k+1}$ using Equation (21) and a backward line search
        $k = k + 1$
    end while
    Calculate the covariance $\mathbf{C}$ according to Equation (22)
    return $\hat{\mathbf{w}}_k$ and $\mathbf{C}$
end procedure
```

4.1 Cantilever Beam Example

The method is first demonstrated for the theoretical Saint-Venant cantilever beam as studied. Assuming plane stress, the Saint-Venant approximation to the strain field is [3]:

$$\boldsymbol{\mathcal{E}}(\mathbf{x}) = \begin{bmatrix} \frac{P}{EI}(L-x)y \\ -\frac{(1+\nu)P}{2EI}\left(\left(\frac{h}{2}\right)^2 - y^2\right) \\ -\frac{\nu P}{EI}(L-x)y \end{bmatrix}, \tag{24}$$

where the geometry is defined in Figure 2. A synthetic strain-free lattice spacing field is defined by

$$d_0(\mathbf{x}) = c_0 \exp\left(-\frac{1}{2}(x-c_1)^2/c_2^2 - \frac{1}{2}(y-c_3)^2/c_4^2\right) + c_5, \tag{25}$$

with the parameters given by $\{c_0, c_1, c_2, c_3, c_4, c_5\} = \{0.0168, 0, 7.5 \times 10^{-3}, 7 \times 10^{-3}, 6 \times 10^{-3}, 4.056\}$. The maximum variation c_0 from a constant base value, c_5, was chosen to reflect the possible maximum relative variation due to martensitic phase change in 0.8% carbon steel.

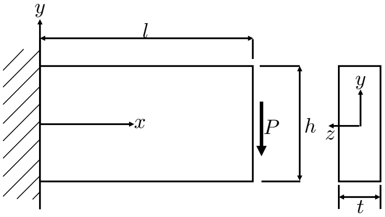


Figure 2: Cantilever beam geometry and coordinate system with $l = 20\text{mm}$, $h = 10\text{mm}$, $t = 5\text{mm}$, $E = 200\text{GPa}$, $P = 2\text{kN}$, $\nu = 0.28$, and $I = \frac{th^3}{12}$.

Measurements of the form (3) were simulated for 30 angles evenly spaced between 0° and 180°, with 100 measurements per angle, which is on the conservative side based on past experiments [8, 12]. The simulated measurements were corrupted with zero-mean noise of standard deviation $\sigma_n = c_5 \times 10^{-4}$ which is equilvalent to 1×10^{-4} standard deviation in strain representing the typical experimental noise [8, 12]

Fifty zero-traction measurements were added along the top and bottom of the cantilever beam for both the presented method and the linear GP regression method. Results are shown in Figure 3. These results show that the presented method successfully reconstructs both the strain field and the d_0 field with a relative error[2] of 0.0057. By contrast ignoring the presence of a d_0 variation and using a linear GP regression method yields a drastically degraded strain reconstruction with a relative error of 0.3067.

4.2 In-situ Loaded C-shape Sample

The method is now demonstrated on a more complex strain field given by FEA of a mild steel C-shape sample with geometry defined in Figure 4. The sample was subjected to a 7 kN compressive load distributed over 5° arcs and plane stress was assumed for the analysis. The resulting FEA strain field is shown in Figure 5. This sample and loading conditions correspond to the experimental setup used by [12].

Measurements of the form (3) were simulated through the this strain and a synthetic, smoothly changing d_0 field is again defined by Equation (25) with parameters given by $\{c_0, c_1, c_2, c_3, c_4, c_5\} = \{0.01, 5 \times 10^{-3}, 7.5 \times 10^{-3}, 7 \times 10^{-3}, 6 \times 10^{-3}, 4.056\}$. A total of 60 strain images were simulated with angles evenly spaced between 0° and 180°, and 180 measurements per image. The simulated measurements were corrupted with zero-mean noise of standard deviation $\sigma_n = c_5 \times 10^{-4}$ which is equivalent to 1×10^{-4} standard deviation in strain representing the typical experimental noise. A total of 131 zero-traction

[2] relative error $= \frac{\text{mean}(|\text{true}-\text{estimated}|)}{\max(|\text{true}|)}$

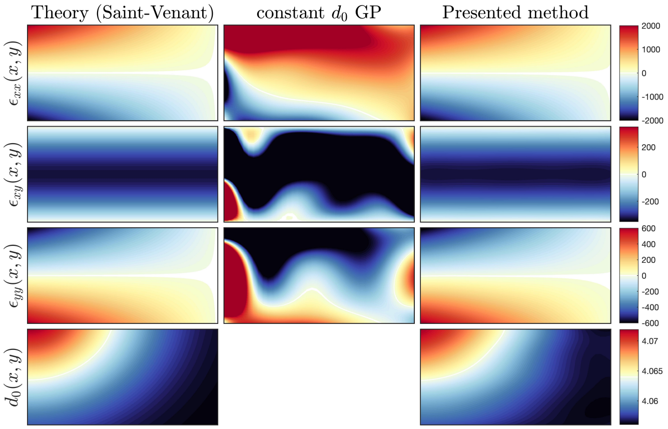


Figure 3: Simulation results for the cantilever beam strain field. The estimated strain field using the presented method is shown as well as the results of assuming a constant d_0 and applying standard GP regression. In the case of the presented method, the estimated d_0 field is also shown. Strain values are given in μStrain.

measurements were added around the boundary of the C-shape excluding the regions within 10° of the loading points. Reconstruction from the LRT and traction measurements was performed using both the presented method and the linear GP regression method, and the results are shown in Figure 5.

The presented method achieves a mean relative error of 0.023 and it can be seen that the reconstruction has achieved the correct shape. Whereas assuming a constant d_0 value gives a mean relative error of 0.137 and the resulting strain fields show incorrect concentrated peaks in the strain field and areas of tension and compression that are reversed. Despite this improvement

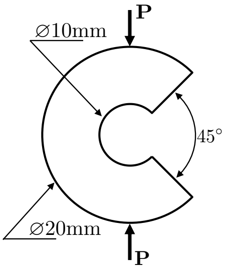


Figure 4: Geometry of the C-shape sample and in-situ loading **P**. The sample has an outer diameter of 20 mm and an inner diameter of 10 mm with a 45° segment removed. The sample was defined to have $E = 200\,\text{GPa}$ and $\nu = 0.28$.

there is still some observable difference. In particular, the presented method has a concentrated tensile region on the top left boundary of the C, and does not capture the very concentrated peaks in magnitude on the inside of the C. These peak strains on the boundary are the hardest for the algorithm to reconstruct as they are poorly sampled by the LRT; i.e. they make up only a very small part of each line integral. Additionally, some of this remaining error is due to systematic error in the simulation of the measurements. Which are generated by numerically performing a line integral with each function evaluation being given by an interpolation of the FEA results.

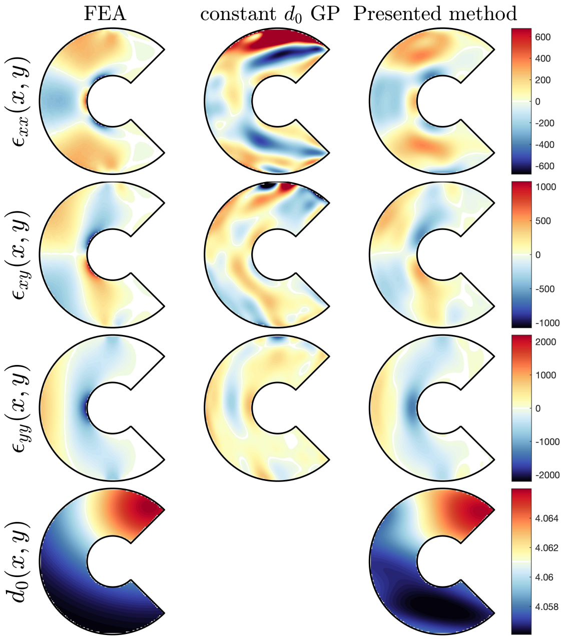


Figure 5: Simulation results for the FEA C-shape sample strain field. Strain values are given as μStrain. The estimated strain field using the presented method is shown as well as the results of assuming a constant d_0 and applying standard GP regression. In the case of the presented method, the estimated d_0 field is also shown.

5 Additional Remarks

5.1 Sensitivity to the Traction Measurement Variance

Boundary conditions given by unloaded surfaces are a natural inclusion as they are an artefact of the physical world. This information is included in the form of artificial measurements of zero traction, however it was found that a small variance needed to be placed on these measurements and the rate of convergence was impacted by the size of this variance. Conceptually, this variance is analogous to a constraint tolerance for optimisation procedures. Too large a variance (or not enough traction measurements) and the algorithm may fail to converge to the correct strain field. This indicates that the traction measurements are ensuring that the problem is observable, which is supported by the findings of [11] where the inclusion of traction measurements allowed a constant d_0 value to be found as a hyperparameter. Conversely, too small a variance and the algorithm is unable to take optimisation steps of significant size, resulting in a large number of iterations to converge. Methods for optimally choosing this variance is an avenue for future research.

Despite this, it was found that the algorithm worked well over a reasonable range of traction variances. Typically the standard deviation of the traction measurements could be set two orders of magnitude smaller than the measurement standard deviation or in the range of 1×10^{-5} to 1×10^{-7}.

5.2 Hyperparameter Optimisation

The hyperparameters $\theta = \{l_x, l_y, \sigma_f\}$ can be found by performing an optimisation using $\mathcal{F}$ as the objective function. However, the gradients of $\mathcal{F}$ with respect to the GP hyperparameters are not trivial and so [30] suggests that gradient free optimisation approaches could be used. In this work, both Bayesian optimisation [20] and the Nelda-Mead method [22] were found to work; with the Nelda-Mead method requiring less computation time.

5.3 Computational cost

The majority of the computational burden comes from building the building the forward model of the LRT measurements. In particular, all the combinations of the basis function of the strain-free lattice spacing and the strain field need to be computed for each measurement. The resulting matrix has $nm_{d_0}m_{\varphi}$ elements. However, as this matrix only needs to be computed once, it is still feasible to solve even with a large number of basis functions.

6 Conclusion

This paper considers an extension of the strain tomography problem where the strain-free lattice parameter is a known constant, to the more general case where it is unknown and varies throughout the sample. A method for the joint reconstruction of a strain field and a varying strain-free lattice parameter from a set of neutron transmission strain images has been presented. This method extends the Gaussian process based approach previously used for strain tomography to the subsequently non-linear problem, and ensures that the estimated strain fields satisfy equilibrium and can include knowledge of boundary conditions. This was achieved by reformulating the problem in terms of basis functions and unknown coefficients. Variational inference was then employed to find estimates of the coefficients.

The method was tested on a set of simulated data, and importantly, these results demonstrate that it is possible to perform this joint reconstruction. Further, the results obtained by ignoring variations in d_0 and applying the linear GP regression method are provided and show that this assumption, if incorrect, severely degrades the accuracy of reconstruction.

Future work will involve planning an experiment to acquire a data set on which to further evaluate the methods performance.

A Basis Functions

The basis functions for the strain field were defined in (13) as

$$\phi_{\epsilon,j} = \begin{bmatrix} \frac{\partial^2}{\partial y^2} - \nu \frac{\partial^2}{\partial x^2} \\ -(1+\nu)\frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial x^2} - \nu \frac{\partial^2}{\partial y^2} \end{bmatrix} \phi_{\varphi,j}. \tag{26}$$

As such, the components of $\phi_{\epsilon,j}$ can be built from

$$\begin{aligned} \frac{\partial^2}{\partial x^2}\phi_{\varphi,j} &= \frac{-\lambda_{\varphi x,j}^2}{\sqrt{L_{\varphi x}L_{\varphi y}}} \sin(\lambda_{\varphi x,j}(x+L_{\varphi x}))\sin(\lambda_{\varphi y,j}(y+L_{\varphi y})), \\ \frac{\partial^2}{\partial y^2}\phi_{\varphi,j} &= \frac{-\lambda_{\varphi y,j}^2}{\sqrt{L_{\varphi x}L_{\varphi y}}} \sin(\lambda_{\varphi x,j}(x+L_{\varphi x}))\sin(\lambda_{\varphi y,j}(y+L_{\varphi y})), \\ \frac{\partial^2}{\partial x \partial y}\phi_{\varphi,j} &= \frac{\lambda_{\varphi x,j}\lambda_{\varphi y,j}}{\sqrt{L_{\varphi x}L_{\varphi y}}} \cos(\lambda_{\varphi x,j}(x+L_{\varphi x}))\cos(\lambda_{\varphi y,j}(y+L_{\varphi y})). \end{aligned} \tag{27}$$

A predicted LRT measurement was defined by (14) as

$$\begin{aligned} y_* = \frac{1}{L}\int_0^L \bar{\mathbf{n}} \left(\sum_j \sum_k \phi_{\epsilon,j}(\mathbf{p}+\hat{\mathbf{n}}s) w_{\varphi,j} \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s) w_{d_0,k} \right) \\ + \left(\sum_k \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s) w_{d_0,k} \right) \mathrm{d}s, \end{aligned} \tag{28}$$

where for clarity we restrict ourselves to a single measurement. Therefore, we need the components

$$\begin{aligned}
&\int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\,\mathrm{d}s,\\
&\int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\left(\frac{\partial^2}{\partial x^2}\phi_{\varphi,j}(\mathbf{p}+\hat{\mathbf{n}}s)\right)\mathrm{d}s,\\
&\int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\left(\frac{\partial^2}{\partial y^2}\phi_{\varphi,j}(\mathbf{p}+\hat{\mathbf{n}}s)\right)\mathrm{d}s,\\
&\int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\left(\frac{\partial^2}{\partial x\partial x}\phi_{\varphi,j}(\mathbf{p}+\hat{\mathbf{n}}s)\right)\mathrm{d}s.
\end{aligned} \tag{29}$$

To make the expressions briefer, we introduce the notation

$$\begin{aligned}
\alpha_{\varphi x} &= \lambda_{\varphi x,j}(x_0+\hat{\mathbf{n}}_1 s+L_{\varphi x}), \quad \alpha_{\varphi y} = \lambda_{\varphi y,j}(y_0+\hat{\mathbf{n}}_2 s+L_{\varphi y}),\\
\alpha_{d_0 x} &= \lambda_{d_0 x,k}(x_0+\hat{\mathbf{n}}_1 s+L_{d_0 x}), \quad \alpha_{d_0 y} = \lambda_{d_0 y,k}(y_0+\hat{\mathbf{n}}_2 s+L_{d_0 y}).
\end{aligned} \tag{30}$$

Giving

$$\begin{aligned}
\zeta_k =& \int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\,\mathrm{d}s\\
=& \frac{1}{\sqrt{L_{d_0x}L_{d_0y}}}\left(\frac{\sin(\alpha_{d_0x}-\alpha_{d_0y})}{2(\hat{\mathbf{n}}_1\lambda_{d_0x,k}-\hat{\mathbf{n}}_2\lambda_{d_0y,k})}-\frac{\sin(\alpha_{d_0x}+\alpha_{d_0y})}{2(\hat{\mathbf{n}}_1\lambda_{d_0x,k}+\hat{\mathbf{n}}_2\lambda_{d_0y,k})}\right)\Bigg|_{s=0}^{s=L},\\
\psi_{1,kj} =& \int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\left(\frac{\partial^2}{\partial x^2}\phi_{\varphi,j}(\mathbf{p}+\hat{\mathbf{n}}s)\right)\mathrm{d}s\\
=& \frac{-\lambda_{\varphi x,j}^2}{\sqrt{L_{\varphi x}L_{\varphi y}L_{d_0x}L_{d_0y}}}\left(-\Gamma_1-\Gamma_2+\Gamma_3+\Gamma_4-\Gamma_5+\Gamma_6-\Gamma_7+\Gamma_8\right)\Bigg|_{s=0}^{s=L},\\
\psi_{2,kj} =& \int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\left(\frac{\partial^2}{\partial y^2}\phi_{\varphi,j}(\mathbf{p}+\hat{\mathbf{n}}s)\right)\mathrm{d}s\\
=& \frac{-\lambda_{\varphi y,j}^2}{\sqrt{L_{\varphi x}L_{\varphi y}L_{d_0x}L_{d_0y}}}\left(-\Gamma_1-\Gamma_2+\Gamma_3+\Gamma_4-\Gamma_5+\Gamma_6-\Gamma_7+\Gamma_8\right)\Bigg|_{s=0}^{s=L},\\
\psi_{3,kj} =& \int_0^L \phi_{d_0,k}(\mathbf{p}+\hat{\mathbf{n}}s)\left(\frac{\partial^2}{\partial x\partial y}\phi_{\varphi,j}(\mathbf{p}+\hat{\mathbf{n}}s)\right)\mathrm{d}s
\end{aligned}$$

$$= \frac{\lambda_{\varphi x,j}\lambda_{\varphi x,j}}{\sqrt{L_{\varphi x}L_{\varphi y}L_{d_0 x}L_{d_0 y}}}\left(-\Gamma_1+\Gamma_2-\Gamma_3+\Gamma_4-\Gamma_5+\Gamma_6+\Gamma_7-\Gamma_8\right)\Bigg|_{s=0}^{s=L}, \tag{31a}$$

where

$$\begin{aligned}
\Gamma_1 &= \frac{\sin(\alpha_{d_0x}-\alpha_{\varphi x}+\alpha_{d_0y}+\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}-\hat{\mathbf{n}}_1\lambda_{\varphi x,j}+\hat{\mathbf{n}}_2\lambda_{d_0y,k}+\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_2 &= \frac{\sin(\alpha_{d_0x}-\alpha_{\varphi x}-\alpha_{d_0y}-\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}-\hat{\mathbf{n}}_1\lambda_{\varphi x,j}-\hat{\mathbf{n}}_2\lambda_{d_0y,k}-\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_3 &= \frac{\sin(\alpha_{d_0x}-\alpha_{\varphi x}+\alpha_{d_0y}-\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}-\hat{\mathbf{n}}_1\lambda_{\varphi x,j}+\hat{\mathbf{n}}_2\lambda_{d_0y,k}-\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_4 &= \frac{\sin(\alpha_{d_0x}+\alpha_{\varphi x}-\alpha_{d_0y}-\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}+\hat{\mathbf{n}}_1\lambda_{\varphi x,j}-\hat{\mathbf{n}}_2\lambda_{d_0y,k}-\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_5 &= \frac{\sin(\alpha_{d_0x}+\alpha_{\varphi x}+\alpha_{d_0y}-\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}+\hat{\mathbf{n}}_1\lambda_{\varphi x,j}+\hat{\mathbf{n}}_2\lambda_{d_0y,k}-\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_6 &= \frac{\sin(\alpha_{d_0x}-\alpha_{\varphi x}-\alpha_{d_0y}+\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}-\hat{\mathbf{n}}_1\lambda_{\varphi x,j}-\hat{\mathbf{n}}_2\lambda_{d_0y,k}+\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_7 &= \frac{\sin(\alpha_{d_0x}+\alpha_{\varphi x}-\alpha_{d_0y}+\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}+\hat{\mathbf{n}}_1\lambda_{\varphi x,j}-\hat{\mathbf{n}}_2\lambda_{d_0y,k}+\hat{\mathbf{n}}_2\lambda_{\varphi y,j})},\\
\Gamma_8 &= \frac{\sin(\alpha_{d_0x}+\alpha_{\varphi x}+\alpha_{d_0y}+\alpha_{\varphi y})}{8(\hat{\mathbf{n}}_1\lambda_{d_0x,k}+\hat{\mathbf{n}}_1\lambda_{\varphi x,j}+\hat{\mathbf{n}}_2\lambda_{d_0y,k}+\hat{\mathbf{n}}_2\lambda_{\varphi y,j})}.
\end{aligned} \tag{31b}$$

Returning to the measurement model in (14), we can now write

$$y_* = \frac{1}{L}\left(\sum_k\sum_j \bar{\mathbf{n}} w_{d_0,k}w_{\varphi j}\begin{bmatrix}\psi_{2,kj}-\nu\psi_{1,kj}\\ -(1+\nu)\psi_{3,kj}\\ \psi_{1,kj}-\nu\psi_{2,kj}\end{bmatrix}_{s=0}^{s=L}+\sum_k w_{d_0,k}\zeta_k\Big|_{s=0}^{s=L}\right). \tag{31c}$$

B Measurement Model Derivatives

Here we give the derivates of the measurement model $g(\cdot)$ about the current $\hat{\mathbf{w}}$. The measurement model is a concatenation of equations (14) and (15), and so we require the derivatives of the predicted LRT measurement, $\hat{\mathbf{y}}$, and the predicted traction $\hat{\mathbf{y}}_t$. For clarity we restrict the following to a single $\hat{\mathbf{y}}$

and $\hat{\mathbf{y}}_t$. The first derivatives are given by

$$\begin{aligned}
\frac{\partial \hat{\mathbf{y}}}{\partial w_{d_0,k}} &= \frac{1}{L}\left(\sum_j \bar{\mathbf{n}} w_{\varphi*,j} \begin{bmatrix} \psi_{2,kj} - \nu\psi_{1,kj} \\ -(1+\nu)\psi_{3,kj} \\ \psi_{1,kj} - \nu\psi_{2,kj} \end{bmatrix}_{s=0}^{s=L} + \zeta_k\Big|_{s=0}^{s=L}\right), \\
\frac{\partial \hat{\mathbf{y}}}{\partial w_{\varphi,k}} &= \frac{1}{L}\left(\sum_k \bar{\mathbf{n}} w_{d_0*,k} \begin{bmatrix} \psi_{2,kj} - \nu\psi_{1,kj} \\ -(1+\nu)\psi_{3,kj} \\ \psi_{1,kj} - \nu\psi_{2,kj} \end{bmatrix}_{s=0}^{s=L}\right), \\
\frac{\partial \hat{\mathbf{y}}_t}{\partial \mathbf{w}_{d_0}} &= 0, \\
\frac{\partial \hat{\mathbf{y}}_t}{\partial \mathbf{w}_{\varphi}} &= \mathbf{T}\boldsymbol{\phi}_{\varphi}(\mathbf{x}).
\end{aligned} \tag{32}$$

The second derivates are

$$\begin{aligned}
\frac{\partial^2 \hat{\mathbf{y}}_t}{\partial \mathbf{w}^2} &= 0, \\
\frac{\partial^2 \hat{\mathbf{y}}}{\partial \mathbf{w}^2} &= \begin{bmatrix} 0 & \frac{\partial^2 \hat{\mathbf{y}}}{\partial \mathbf{w}_{d_0} \partial \mathbf{w}_{\varphi}} \\ \left(\frac{\partial^2 \hat{\mathbf{y}}}{\partial \mathbf{w}_{d_0} \partial \mathbf{w}_{\varphi}}\right)^{\mathsf{T}} & 0 \end{bmatrix},
\end{aligned} \tag{33}$$

where

$$\left[\frac{\partial^2 \hat{\mathbf{y}}}{\partial \mathbf{w}_{d_0} \partial \mathbf{w}_{\varphi}}\right]_{kj} = \bar{\mathbf{n}} \begin{bmatrix} \psi_{2,kj} - \nu\psi_{1,kj} \\ -(1+\nu)\psi_{3,kj} \\ \psi_{1,kj} - \nu\psi_{2,kj} \end{bmatrix}_{s=0}^{s=L}. \tag{34}$$

Explicit formulation of the second derivatives allows the cost functions curvature to be taken into account in the optimisation procedure, greatly improving the rate of convergence.

References

[1] B. Abbey, S. Zhang, W. Vorster and A. Korsunsky. “Feasibility study of neutron strain tomography”. In: *Procedia Engineering* 1.1 (2009), pp. 185–188.

[2] B. Abbey, S. Zhang, W. Vorster and A. Korsunsky. “Reconstruction of axisymmetric strain distributions via neutron strain tomography”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 270 (2012), pp. 28–35.

[3] F. Beer, E. Johnston Jr, J. Dewolf and D. Mazurek. *MECHANICS OF MATERIALS, sixth edit Edition.* 2010.

[4] C. Bishop et al. *Neural networks for pattern recognition.* Oxford university press, 1995.

[5] D. Blei, A. Kucukelbir and J. McAuliffe. “Variational inference: A review for statisticians”. In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877.

[6] W. Choi, L. Li, V. Luzin, R. Neiser, T. Gnäupel-Herold, H. Prask, S. Sampath and A. Gouldstone. “Integrated characterization of cold sprayed aluminum coatings”. In: *Acta Materialia* 55.3 (2007), pp. 857–866.

[7] A. Gregg, J. Hendriks, C. Wensrich and M. Meylan. “Tomographic reconstruction of residual strain in axisymmetric systems from Bragg-edge neutron imaging”. In: *Mechanics Research Communications* 85 (2017), pp. 96–103.

[8] A. Gregg, J. Hendriks, C. Wensrich, A. Wills, A. Tremsin, V. Luzin, T. Shinohara, O. Kirstein, M. Meylan and E. Kisi. “Tomographic reconstruction of two-dimensional residual strain fields from bragg-edge neutron imaging”. In: *Physical Review Applied* 10.6 (2018), p. 064034.

[9] J. Hendriks. *Joint tomography of strain and the strain-free lattice spacing from neutron transmission strain images.* `https://github.com/jnh277/Joint_strain_d0_tomography.git`. 2019.

[10] J. Hendriks, A. Gregg, C. Wensrich and A. Wills. *Implementation of Traction Constraints in Bragg-edge Neutron Transmission Strain Tomography.* Tech. rep. arXiv preprint arXiv:1805.09760, May 2018.

[11] J. Hendriks, C. Wensrich, A. Wills, V. Luzin and A. T. Gregg. *Robust Inference of Two-Dimensional Strain Fields from Diffraction-based.* Tech. rep. arXiv preprint arXiv:1808.06282, 2018.

[12] J. N. Hendriks, A. W. T. Gregg, C. M. Wensrich, A. S. Tremsin, T. Shinohara, M. Meylan, E. H. Kisi, V. Luzin and O. Kirsten. “Bragg-edge elastic strain tomography for in situ systems from energy-resolved neutron transmission imaging”. In: *Physical Review Materials* 1.5 (Oct. 2017), p. 053802.

[13] P. Hennig and M. Kiefel. “Quasi-Newton methods: A new direction”. In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 843–865.

[14] C. Jidling, J. Hendriks, N. Wahlström, A. Gregg, T. B. Schön, C. Wensrich and A. Wills. “Probabilistic modelling and reconstruction of strain”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 436 (2018), pp. 141–155.

[15] C. Jidling, N. Wahlström, A. Wills and T. B. Schön. “Linearly constrained Gaussian processes”. In: *Advances in Neural Information Processing Systems (NIPS).* Los Angeles, US, 2017, pp. 1215–1224.

[16] M. Jordan, Z. Ghahramani, T. Jaakkola and L. Saul. “An introduction to variational methods for graphical models”. In: *Machine learning* 37.2 (1999), pp. 183–233.

[17] H. Kirkwood, S. Zhang, A. Tremsin, A. Korsunsky, N. Baimpas and B. Abbey. “Neutron Strain Tomography using the Radon Transform”. In: *Materials Today: Proceedings* 2 (2015), S414–S423.

[18] W. R. B. Lionheart and P. J. Withers. “Diffraction tomography of strain”. In: *Inverse Problems* 31.4 (2015), p. 045005.

[19] V. Luzin, K. Spencer and M.-X. Zhang. “Residual stress and thermomechanical properties of cold spray metal coatings”. In: *Acta Materialia* 59.3 (2011), pp. 1259–1270.

[20] J. Mockus. *Bayesian approach to global optimization: theory and applications.* Vol. 37. Springer Science & Business Media, 2012.

[21] I. Murray and R. Adams. “Slice sampling covariance hyperparameters of latent Gaussian models”. In: *Advances in neural information processing systems.* 2010, pp. 1732–1740.

[22] J. Nelder and R. Mead. “A simplex method for function minimization”. In: *The computer journal* 7.4 (1965), pp. 308–313.

[23] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes.* New York: McGraw-Hill Education, 1991.

[24] A. Paradowska, J. Price, R. Ibrahim and T. Finlayson. “A neutron diffraction study of residual stress due to welding”. In: *Journal of materials processing technology* 164 (2005), pp. 1099–1105.

[25] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning.* MIT press, Cambridge, MA, 2006.

[26] M. Sadd. *Elasticity: theory, applications, and numerics.* Academic Press, 2009.

[27] J. R. Santisteban, L. Edwards, M. E. Fitzpatrick, A. Steuwer, P. J. Withers, M. Daymond, M. W. Johnson, N. Rhodes and E. M. Schooneveld. “Strain imaging by Bragg edge neutron transmission”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 481.1 (2002), pp. 765–768.

[28] J. R. Santisteban, L. Edwards, M. E. Fizpatrick, A. Steuwer and P. J. Withers. “Engineering applications of Bragg-edge neutron transmission”. In: *Applied Physics* 74.1 (2002), pp. 1433–1436.

[29] A. Solin and S. Särkkä. “Hilbert space methods for reduced-rank Gaussian process regression”. In: *Statistics and Computing* (Aug. 2019).

[30] D. Steinberg and E. Bonilla. “Extended and unscented Gaussian processes”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 1251–1259.

[31] A. Tremsin, J. McPhate, W. Kockelmann, J. Vallerga, O. H. W. Siegmund and W. B. Feller. “High resolution Bragg edge transmission spectroscopy at pulsed neutron sources: Proof of principle experiments with a neutron counting MCP detector”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors, and Associated Equipment* 633 (May 2011), S235–S238.

[32] A. Tremsin, J. McPhate, A. Steuwer, W. Kockelmann, A. Paradowska, J. Kelleher, J. Vallerga, O. H. W. Siegmund and W. B. Feller. “High-Resolution Strain Mapping Through Time-of-Flight Neutron Transmission Diffraction with a Microchannel Plate Neutron Counting Detector”. In: *Strain* 48 (Aug. 2012).

[33] N. Wahlström. “Modeling of Magnetic Fields and Extended Objects for Localization Applications”. PhD thesis. Division of Automatic Control, Linköping University, 2015.