Scalable Magnetic Field Modeling Using Structured Kernel Interpolation for Gaussian Process Regression

M.P.D. Fetter

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Scalable Magnetic Field Modeling Using Structured Kernel Interpolation for Gaussian Process Regression

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

M.P.D. Fetter

January 8, 2023

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Indoor positioning systems cannot rely on conventional localization methods, such as GPS, to locate devices because of interference with the structure of buildings. One solution is to use magnetic positioning, which is based on spatial variations in the patterns of the ambient magnetic field. To model magnetic fields, Gaussian process regression is used, providing predictions of the magnetic field at unvisited locations along with uncertainty quantification. These predictions and their uncertainties are valuable information for probabilistic localization algorithms used for magnetic positioning. Full Gaussian process regression has poor scalability, becoming computationally intractable from roughly 10,000 one-dimensional measurements due to its associated cubic computational complexity. In the existing literature, approximations for Gaussian process regression have been extensively studied to reduce this computational complexity. Of these approximations, only approximations involving basis functions and local experts have been used in the context of scalable magnetic field modeling. A favorable approximation framework from existing literature uses structured kernel interpolation (SKI), allowing for fast regression through efficient Krylov subspace methods. The SKI framework is favorable as it allows for fast regression in low dimensions without introducing boundary effects. In this thesis, the SKI framework is used to approximate two distinct magnetic field models: the shared model, which considers independence between the magnetic field components with shared hyperparameters, and the scalar potential model, which includes physical properties of the magnetic field (Maxwell's equations) in the model. The scalability of the approach is shown using simulations and experiments with magnetic field measurements. Through the simulations, it is shown the SKI framework accurately and efficiently approximates the models. The applicability of the SKI framework for scalable magnetic field modeling is investigated using data collected using a motion capture suit, the Xsens MVN Link Suit. In the final experiment, a magnetic field map is constructed based on more than 40,000 three-dimensional measurements without splitting the data set, which took less than one minute on a standard laptop.

Keywords: Magnetic field modeling, Gaussian processes, structured kernel interpolation (SKI), motion capture suit (Xsens MVN Link Suit).

Table of Contents

List of Figures

List of Tables

Preface

This document contains my thesis concerning scalable magnetic field modeling using the SKI framework for Gaussian process regression. The idea of the thesis topic came after some discussions with my supervisors dr. Manon Kok and Clara Menzen of Delft Center for Systems and Control (DCSC), Delft University of Technology. I had previously worked on modeling magnetic field maps using Gaussian process regression during my bachelor's thesis, where the scalability issues associated with Gaussian process regression posed an issue. During that time, the scalability of Gaussian processes was beyond the scope of the project, as simply downsampling the available data provided sufficient information. The scalability of Gaussian processes for modeling magnetic field maps remained a topic of interest to me, so I decided to further explore this path. Preliminary research led me to the SKI framework, a promising approach for scalable magnetic field modeling.

I would like to thank my daily supervisor Clara Menzen for her ideas and guidance during the thesis, as well as her enthusiasm during the biweekly meetings. I would also like to thank my supervisor dr. Manon Kok for her feedback and guidance, which helped improve the quality of the thesis. Additionally, I want to thank Fabian Girrbach from Xsens for extracting the necessary data from the data collected with the motion capture suit. Lastly, I would like to thank fellow (graduated) student Thijs Veen for his help with data collection and for providing me with the data set used in the motion capture lab experiment.

Delft, University of Technology — M.P.D. Fetter
January 8, 2023

Chapter 1

Introduction

Over the recent years, indoor positioning systems have been researched extensively [1, 2]. These systems are used to accurately locate devices in (large) indoor areas, such as shopping malls, airports, parking garages, etc. In modern devices, such as smartphones or automotive systems, location services use (combinations of) many different sensors and systems to position the device. Examples include GPS, cellular networks, inertial sensors, optical sensors (e.g., cameras, LiDAR), Bluetooth, and Wi-Fi [3, 4]. For indoor positioning, however, the use of these systems may pose issues. A widely recognized drawback associated with using GPS and cellular networks for indoor localization is the lack of accuracy indoors, due to signal interference with the material of the structure of buildings [5]. Dead reckoning techniques which only rely on the measurements of inertial sensors lack accuracy over time due to cumulative errors [6]. The other methods require additional infrastructure or an undisturbed line of sight, which may not be present.

One particular solution for indoor localization without required external hardware proposed in literature is based on the presence of the ambient magnetic field [5, 7]. Ferromagnetic materials in the structure of indoor locations and objects lead to anomalies in the magnetic field [2]. These anomalies result in magnetic field patterns throughout an indoor area [7], which can be modeled and visualized in a map. Models of the magnetic field are a crucial step for indoor localization and simultaneous localization and mapping (SLAM) algorithms, which are often based on extended Kalman filters [8] or particle filters [9]. Localization algorithms require a detailed map of the indoor magnetic field, whereas SLAM algorithms (additionally) need a method for the development of magnetic field maps.

An established approach for magnetic field modeling is Gaussian process regression [10, 11, 12, 13, 14]. Gaussian processes are a powerful tool for statistical inference of continuous variables, such as the magnetic field. Interpolation at unvisited locations is performed with observed measurement data, assuming a non-linear model function over the domain of interest. In the case of magnetic fields, the measurements of the magnetic field are considered to be the outputs, whereas the positional data associated with the measurements are the inputs. In practice, when modeling magnetic fields, a two- or three-dimensional input space is used for Gaussian process regression. Two-dimensional models generally require less data for a

satisfactory model due to the reduced dimensionality, whereas three-dimensional models can contain more information. A valuable property of Gaussian processes is that they can handle noise present in the measurements, which occurs naturally when measuring the magnetic field using magnetometers. Additionally, Gaussian process regression gives a quantification of the uncertainty of predicted variables. This is valuable information for probabilistic localization algorithms, such as extended Kalman filters and particle filters, as it can be used to assign weights to the measurements.

A significant drawback of Gaussian process regression is its (lack of) scalability [15]. Full Gaussian process regression has an associated cubic computational complexity, $\mathcal{O}(N^3)$ meaning the time required to perform inference scales cubically with the number of measurements N. This typically poses issues on a standard computer from 10,000 measurements and above. As the use of more magnetic field measurements means more information is used to estimate the magnetic field, it is desirable to be able to exceed this threshold. The scalability of Gaussian processes through approximations is a topic of large interest and has been researched extensively over the past decades [15]. Some of these approximations may be used for magnetic field modeling depending on their associated properties.

In previous research, the Laplace operator eigenbasis approximation for Gaussian process regression has been developed for scalable magnetic field modeling, which is based on the use of basis functions to approximate the kernel function characterizing the predictions [14, 16]. The approach uses the solutions to the negative Laplace equations as the basis functions on a fixed domain, reducing the computational complexity associated with Gaussian process regression to $\mathcal{O}(NM_{\text{bf}}^2)$, where $M_{\text{bf}} < N$ is the number of basis functions, thus allowing more measurements to be used. A downside of this approach is the presence of boundary conditions, reverting the predictions and their corresponding uncertainties to the prior near the domain boundaries. The Laplace operator eigenbasis approach has been used in various magnetic field SLAM applications [8, 17, 18].

Another favorable approach described in literature for scalable Gaussian process regression is the structured kernel interpolation (SKI) framework. In this framework, the measurements are observed through M latent variables, called the inducing variables. By forcing these inducing variables into a Cartesian grid structure, efficient mathematical formulations – most notably Krylov subspace methods – may be exploited within this framework for the reduction of the associated computational complexity [19]. The computational complexity for three-dimensional magnetic field modeling using the SKI framework is $\mathcal{O}(K(N + 3M^{4/3}))$ [19, 20], where $K \ll N, M$ is based on the desired characteristics of the Krylov subspace methods. This computational complexity is smaller than that of the Laplace operator eigenbasis approach, although the accuracy of the two approximations is not directly proportionate based on the numbers of inducing points M and basis functions M_{bf}. Additionally, the SKI framework does not introduce a reversion of the predictions and their uncertainties to the prior near the domain boundaries. Due to the favorable computational complexity associated with the SKI framework (for low dimensions, $D < 5$) and the lack of boundary conditions near the domain boundaries, the SKI framework is a promising approach for scalable magnetic field modeling.

In this thesis, a scalable approach for magnetic field modeling is developed based on the SKI framework as an alternative to the existing Laplace operator eigenbasis approach. The SKI framework is integrated into two models for magnetic field estimation from existing literature: the shared model and the scalar potential model. The shared model assumes

independence between the magnetic field components with shared hyperparameters to prevent large behavioral differences [14]. The scalar potential model includes Maxwell's equations governing the magnetic field in the model by modeling the magnetic field as the negative gradient of a latent scalar potential function [13]. The following research question is considered throughout this thesis: *How can scalable magnetic field modeling be achieved using the SKI framework for Gaussian process regression?* To answer this question in a structured approach, three subquestions are formulated, given by

i. How can the SKI framework be integrated in the shared and scalar potential models for scalable magnetic field estimation?

ii. How does the SKI framework perform for scalable modeling using the shared and scalar potential models in terms of accuracy and efficiency?

iii. How does the SKI framework for Gaussian process regression perform for large-scale magnetic field modeling?

The first subquestion concerns the integration of the SKI framework into the shared and scalar potential models in a mathematical sense, such that the formulations can be used in numerical implementations. For the second and third subquestions, simulations are conducted and experiments are done using real magnetic field measurements. In the experimental setup, real data are collected using a state-of-the-art motion capture suit, the MVN Link Suit (Xsens Technologies B.V.), containing 17 inertial measurement units (IMUs) equipped with magnetometers distributed over the body. The MVN Link Suit is used as the suit and its accompanying software provide the magnetic field measurements with position and orientation estimates of these 17 sensors at 240 Hz, without requiring additional infrastructure. The position estimates are beneficial as they are used as the inputs for Gaussian process regression, whereas the orientation measurements are used to rotate the magnetic field measurements to a consistent global frame. While it cannot be expected the position estimates are ground truth, it is assumed no noise is present in the position estimates to prevent the models from becoming too complex for approximation using the SKI framework. It should be noted that the use of magnetic field measurements of multiple sensors does not necessarily lead to better estimation of the magnetic field [21]. Inconsistencies occurring due to improper calibration of the magnetometers or improper calibration of the body proportions of the person wearing the suit may be propagated into the training data and thus the predictions. To avoid inconsistencies from occurring, the measurements of only one sensor are used for experimental results. The use of the suit is still beneficial, as the position and orientation estimates of the sensor are computed based on the data of all sensors without requiring additional infrastructure.

The organization of the thesis is based on the research question and the order of its subquestions. First, related work containing beneficial background information on Gaussian process regression, magnetic fields, and the MVN Link Suit is presented in Chapter 2. Integration of the SKI framework in the shared and scalar potential models for magnetic field estimation and the experimental setup are described in Chapter 3. In Chapter 4, the accuracy and associated efficiency of the approximated models are investigated through simulations and the experimental setup. Additionally, large-scale experiments are conducted in this chapter, demonstrating the scalability of magnetic field modeling using the SKI framework for Gaussian process regression. Lastly, the results are discussed and conclusions are drawn in Chapter 5, along with suggestions for further research.

Mathematical Notation

To ensure all variables used in this thesis are understood correctly, Table 1-1 provides an overview of the notations used. Throughout the thesis, the domains of vector and matrix variables are presented to differentiate the variable types.

Table 1-1: Overview of the notations used, p, $q > 1$ are arbitrary dimensions.

Description	Notation	Domain
Scalar	g or G	$g \in \mathbb{R}$
Vector	$\mathbf{g}$ or $\mathbf{G}$	$\mathbf{g} \in \mathbb{R}^p$
Matrix	$\mathbf{g}$ or $\mathbf{G}$	$\mathbf{g} \in \mathbb{R}^{p \times q}$
Scalar-valued function	$g(\cdot)$	$g : \mathbb{R}^p \to \mathbb{R}$
Vector-valued function	$\mathbf{g}(\cdot)$	$\mathbf{g} : \mathbb{R}^p \to \mathbb{R}^q$

Chapter 2

Background Information

This chapter provides beneficial background information from related work concerning modeling magnetic fields using Gaussian process regression. This includes full Gaussian process regression, magnetic fields, their combination, and the experimental setup with the motion capture suit. Necessary background information on Gaussian process regression is presented in Section 2-1. Beneficial magnetic field theory is stated in Section 2-2. Modeling magnetic fields using Gaussian process regression is presented in Section 2-3, where the magnetic field theory is taken into account. The characteristics of the MVN Link Suit and its accompanying software are described in Section 2-4.

2-1 Full Gaussian Process Regression

Gaussian process regression is an interpolation method where interpolation is achieved through Bayesian inference with Gaussian processes. Its most basic form is single-output full Gaussian process regression using scalar measurements y_i. The model for this type of regression assumes a non-linear model function $f(\mathbf{x})$ over a set of N measurements, $\mathbf{Y} = [y_1 \quad y_2 \quad \dots \quad y_N]^\top \in \mathbb{R}^N$. The measurements are observed at inputs $\mathbf{X_f} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \dots \quad \mathbf{x}_N]^\top \in \mathbb{R}^{N \times D}$, where $\mathbf{x}_i \in \mathbb{R}^D$ with input dimensionality D. For this thesis, $D = 3$ is considered, corresponding to a three-dimensional measurement space. The model function $f(\mathbf{x})$ is uniquely defined by its covariance function, $\kappa(\mathbf{x}, \mathbf{x}')$, with $\mathbf{f}$ defined as $\mathbf{f} = [f_1 \quad f_2 \quad \dots \quad f_N]^\top \in \mathbb{R}^N$. The measurements are assumed to be corrupted by white Gaussian noise, i.e., $\varepsilon_i \sim \mathcal{N}(0, \sigma_\text{n}^2)$. This model is commonly expressed as

$$\begin{aligned} f(\mathbf{x}) &\sim \mathcal{GP}(0, \kappa(\mathbf{x}, \mathbf{x}')), \\ y_i &= f(\mathbf{x}_i) + \varepsilon_i. \end{aligned} \tag{2-1}$$

The covariance function $\kappa(\mathbf{x}, \mathbf{x}')$, also called the kernel function, describes the covariance between pairs of inputs. This kernel function is used to construct covariance matrices, which describe the covariance between sets of inputs. The (i, j)-th entry of a covariance matrix $\mathbf{K}$ is computed as $\kappa(\mathbf{x}_i, \mathbf{x}_j)$.

Inference is achieved using the predictive distribution. This distribution describes the expected value and variance of the model function at test input $\mathbf{x}_* \in \mathbb{R}^D$. The N_* test inputs are collected in the test input matrix $\mathbf{X}_* \in \mathbb{R}^{N_* \times D}$. The predictive distribution for full Gaussian process regression is given by

$$\mathbb{E}\left[\mathbf{f}(\mathbf{X}_*)\right] = \mathbf{K}_{*,\mathbf{f}} \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \mathbf{Y}, \tag{2-2a}$$

$$\mathbb{V}\left[\mathbf{f}(\mathbf{X}_*)\right] = \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{f}} \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \mathbf{K}_{\mathbf{f},*}. \tag{2-2b}$$

The subscripts of the covariance matrices denote the sets of inputs used. For example, $\mathbf{K}_{*,\mathbf{f}}$ denotes the covariance between the test inputs $\mathbf{X}_*$ and the training inputs $\mathbf{X}_\mathbf{f}$. $\mathbb{E}\left[\mathbf{f}(\mathbf{X}_*)\right]$ denotes the expected value of the predicted model function at test inputs $\mathbf{X}_*$, whereas $\mathbb{V}\left[\mathbf{f}(\mathbf{X}_*)\right]$ denotes the variance of this prediction at the test inputs.

The predictions are heavily influenced by the choice of the kernel function. A commonly used kernel function is the squared exponential kernel, which is an infinitely continuously differentiable and thus smooth function. The squared exponential kernel is well-defined and is commonly expressed as

$$\kappa_\mathrm{SE}(\mathbf{x}, \mathbf{x}') = \sigma_\mathrm{f}^2 \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\ell^2}\right), \tag{2-3}$$

containing two characteristic parameters: the length scale ℓ and the variance σ_f^2. The length scale ℓ describes how much two points influence each other based on the distance between them. The variance σ_f^2 characterizes the average squared distance of the model function away from its mean. The latter is also often interpreted as a scale factor, simply scaling the magnitude of the kernel function. Additionally, a constant kernel function may be added to the squared exponential kernel to allow the model to recognize non-zero means in the data [13], such as a constant deviation in Earth's magnetic field [14]. The constant kernel function is given by

$$\kappa_\mathrm{const.}(\mathbf{x}, \mathbf{x}') = \sigma_\mathrm{c}^2. \tag{2-4}$$

Together with the model noise variance σ_n^2 and the squared exponential kernel parameters, these parameters are called the hyperparameters $\boldsymbol{\theta}$, i.e., $\boldsymbol{\theta} = \begin{bmatrix}\ell & \sigma_\mathrm{f} & \sigma_\mathrm{n} & \sigma_\mathrm{c}\end{bmatrix}^\top$. If one does not wish to add a constant kernel for practical reasons, one can alternatively decide to subtract the mean from the measurements, adding these again after inference [10]. The predictive mean of equation (2-2a) is then expressed as

$$\mathbb{E}\left[\mathbf{f}(\mathbf{X}_*)\right] = \boldsymbol{\mu}(\mathbf{Y}) + \mathbf{K}_{*,\mathbf{f}} \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \left(\mathbf{Y} - \boldsymbol{\mu}(\mathbf{Y})\right), \tag{2-5}$$

where $\boldsymbol{\mu}(\mathbf{Y})$ is the mean of the measurements. This approach is not as complete as adding a constant kernel, because the uncertainty of the estimation of the mean is not included [10].

As the model is fully defined by the hyperparameters $\boldsymbol{\theta}$, a proper choice is required for an accurate representation of the underlying model function. One approach to determining the hyperparameters is to learn them from the available measurements by maximizing the marginal likelihood function. This function quantifies how well a Gaussian process fits the available data [10]. For optimization purposes, the log marginal likelihood function is used,

which, for full Gaussian process regression, is given by

$$\log p(\mathbf{Y}|\boldsymbol{\theta}) = \underbrace{-\frac{1}{2}\mathbf{Y}^\top \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2\mathbf{I}\right)^{-1}\mathbf{Y}}_{\text{data fit}} \underbrace{-\frac{1}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2\mathbf{I}\right|}_{\text{complexity penalty}} - \frac{N}{2}\log 2\pi. \tag{2-6}$$

The log marginal likelihood function contains two important terms: the data fit and the complexity penalty. The data fit ensures the optimized model is accurately represented by the training data, whereas the complexity penalty avoids overfitting to the specific training data [10]. To ensure the optimized squared exponential hyperparameters are positive without constraining the optimization problem, exponentiated versions are substituted during optimization, expressed as

$$\begin{aligned} \exp(l) &= \ell^2, \\ \exp(s_\mathrm{f}) &= \sigma_\mathrm{f}^2, \\ \exp(s_\mathrm{n}) &= \sigma_\mathrm{n}^2. \end{aligned} \tag{2-7}$$

The hyperparameter of the constant kernel, σ_c^2, is not exponentiated, as the constant deviation of Earth's magnetic field may be negative. Solving the optimization problem is done using the hyperparameters $\boldsymbol{\theta}_\mathrm{exp} = [l \quad s_\mathrm{f} \quad s_\mathrm{n} \quad \sigma_\mathrm{c}]^\top$. The problem is generally solved using efficient gradient-based optimization algorithms for unconstrained non-linear problems, such as the conjugate gradient optimization method.

A widely recognized drawback of full Gaussian process regression is its cubic computational complexity with respect to the number of measurements N, i.e., $\mathcal{O}(N^3)$. This complexity occurs due to the inversion of the $\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2\mathbf{I}$ matrix [10]. Common practice is to compute the Cholesky decomposition $\mathbf{L}\mathbf{L}^\top$ of $\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma_\mathrm{n}^2\mathbf{I}$ to solve linear systems involving the matrix. This approach allows for numerically stable solutions and avoids the recomputation of the inverse. Inconveniently, the computation of the Cholesky decomposition retains a computational complexity of $\mathcal{O}(N^3)$. Consequently, full Gaussian process regression is impractical for large data sets ($N > 10,000$).

2-2 Magnetic Fields

For accurate modeling of the magnetic field based on the measurements, magnetic field theory is presented. The magnetic field is a three-dimensional vector field that can be described by two distinct definitions: the **B**-field and the **H**-field. $\mathbf{B} \in \mathbb{R}^3$ is the magnetic flux density, measured in tesla (T), whereas $\mathbf{H} \in \mathbb{R}^3$ is the magnetic field strength, measured in ampere per meter (A/m). The **B**-field and the **H**-field are related through the expression

$$\mathbf{H} \equiv \frac{1}{\mu_0}\mathbf{B} - \mathbf{M}, \tag{2-8}$$

where μ_0 is the vacuum permeability, equal to $1.25663706212 \times 10^{-6}$ N A^{-2}. $\mathbf{M} \in \mathbb{R}^3$ is the magnetization vector, often assumed to be zero in the context of magnetic field modeling, considering the material around the sensor is non-magnetic [13]. This is assumed to hold as the casing of the IMUs of the MVN Link Suit is purposely chosen to be made of plastic, preventing interference with the magnetometer. A second assumption considers the absence of free currents (e.g., wires), meaning Maxwell's equations governing the magnetic field's physics

lead to the statements that the **B**-field is divergence-free and the **H**-field is curl-free [13]. These statements can be expressed as two field equations, given by

$$\nabla \cdot \mathbf{B} = 0, \tag{2-9a}$$
$$\nabla \times \mathbf{H} = \mathbf{0}. \tag{2-9b}$$

Lastly, the assumption that $\mathbf{M} = \mathbf{0}$ leads to the field equations being infinitely continuously differentiable, implying smoothness [13]. This justifies the use of the smooth squared exponential kernel to model small-scale variations in the magnetic field. Considering $\mathbf{M} = \mathbf{0}$, $\mathbf{B}$ and $\mathbf{H}$ are directly related through $\mathbf{H} = \frac{1}{\mu_0}\mathbf{B}$. As the relation scales through the constant μ_0, either of the divergence- and curl-free properties may be exploited.

Throughout this thesis, the curl-free property of the **H**-field is considered for the inclusion of physical knowledge in Gaussian processes. An important property of a curl-free vector field is that it can be written as the negative gradient of a scalar potential, in this case, the magnetic scalar potential φ [13], i.e.,

$$\mathbf{H} = -\nabla\varphi. \tag{2-10}$$

2-3 Modeling Magnetic Fields Using Gaussian Process Regression

Modeling magnetic fields using Gaussian process regression requires a different model than the basic single-output model of equation (2-2). As stated in the previous section, the magnetic field is a vector field, which can be observed through vector-valued measurements $\mathbf{y}_i \in \mathbb{R}^3$. In this thesis, two notable models are considered for magnetic field estimation: the shared model and the scalar potential model, which both have been used in existing literature concerning magnetic field modeling. The shared model assumes independent regression between the three components of $\mathbf{H}$, i.e., $h^{(d)}$ for $d = 1, 2, 3$, with shared hyperparameters to prevent large behavioral differences per component [14]. This model has been previously used in a magnetic field SLAM application [22]. The shared model is presented in Section 2-3-1. The scalar potential model assumes the magnetic field satisfies the curl-free constraint given by equation (2-9b), by modeling the magnetic field as the negative gradient of a latent scalar potential function [13]. It has been used for magnetic field estimation and in various SLAM applications [8, 12, 14, 17]. The scalar potential model for magnetic field modeling is presented in Section 2-3-2.

The N magnetic field measurements used for Gaussian process regression are denoted by the variables $\mathbf{Y} \in \mathbb{R}^{N\times 3}$ and $\mathbf{Y}^{(d)} \in \mathbb{R}^N$, where the superscript $^{(d)}$ denotes the different magnetic field components. The variables are related through

$$\mathbf{Y} = \begin{bmatrix}\mathbf{Y}^{(1)} & \mathbf{Y}^{(2)} & \mathbf{Y}^{(3)}\end{bmatrix} = \begin{bmatrix} y_1^{(1)} & y_1^{(2)} & y_1^{(3)} \\ y_2^{(1)} & y_2^{(2)} & y_2^{(3)} \\ \vdots & \vdots & \vdots \\ y_N^{(1)} & y_N^{(2)} & y_N^{(3)} \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1^\top \\ \mathbf{y}_2^\top \\ \vdots \\ \mathbf{y}_N^\top \end{bmatrix}. \tag{2-11}$$

2-3-1 Shared Modeling of the Magnetic Field

As mentioned previously, the shared model assumes independent regression between the components of the magnetic field with shared hyperparameters, to avoid large behavioral differences per component. This can be modeled using a multi-output Gaussian process with a diagonal kernel function [12]. To allow for different constant deviations of Earth's magnetic field per component, the hyperparameters associated with the constant kernel are not shared between the components. This model is given by

$$\begin{aligned} \mathbf{H}(\mathbf{x}) &\sim \mathcal{GP}(\mathbf{0}, \boldsymbol{\kappa}_{\text{const.}}(\mathbf{x}, \mathbf{x}') + \kappa_{\text{SE}}(\mathbf{x}, \mathbf{x}')\mathbf{I}_3), \\ \mathbf{y}_i &= \mathbf{H}(\mathbf{x}_i) + \mathbf{e}_i, \quad \mathbf{e}_i \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{n}}^2\mathbf{I}_3), \end{aligned} \tag{2-12}$$

where $\boldsymbol{\kappa}_{\text{const.}}(\mathbf{x}, \mathbf{x}')$ is a 3×3 diagonal kernel describing the constant deviations of Earth's magnetic field per component. Due to the diagonal kernel function, this model can also be written as three separate Gaussian processes [14], expressed as

$$\begin{aligned} h^{(d)}(\mathbf{x}) &\sim \mathcal{GP}(0, \kappa_{\text{const.}}^{(d)}(\mathbf{x}, \mathbf{x}') + \kappa_{\text{SE}}(\mathbf{x}, \mathbf{x}')), \\ y_i^{(d)} &= h^{(d)}(\mathbf{x}_i) + \varepsilon_i^{(d)}, \quad \varepsilon_i^{(d)} \sim \mathcal{N}(0, \sigma_{\text{n}}^2). \end{aligned} \tag{2-13}$$

For shared learning of the hyperparameters, a modified log marginal likelihood function is required [14], which is given by

$$\log p(\mathbf{Y}|\boldsymbol{\theta}) = -\frac{1}{2}\sum_{d=1}^{3}\left(\left(\mathbf{Y}^{(d)}\right)^{\top}\left(\mathbf{K}_{\mathbf{f},\mathbf{f}}^{(d)} + \sigma_{\text{n}}^2\mathbf{I}\right)^{-1}\mathbf{Y}^{(d)}\right) - \frac{3}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{f}}^{(d)} + \sigma_{\text{n}}^2\mathbf{I}\right| - \frac{3N}{2}\log 2\pi, \tag{2-14}$$

where $\mathbf{K}_{\mathbf{f},\mathbf{f}}^{(d)}$ changes per component due to the added constant kernel.

For inference with the shared model, the predictive distribution of equation (2-2) is computed for each magnetic field component, requiring the solution to three $N \times N$ systems. As the predictive variances are only dependent on the inputs and not the measurement values, the predictive variances are equal for each component and thus only need to be computed once.

2-3-2 Modeling the Magnetic Field Using a Scalar Potential Function

The scalar potential model for magnetic field modeling considers the curl-free assumption, derived from Maxwell's equations governing the magnetic field's physics. As described in Section 2-2, the assumption of the curl-free constraint means the magnetic field can be written as the negative gradient of the magnetic scalar potential. By considering the magnetic field measurements to be the negative gradients of this scalar potential with added white Gaussian noise, the magnetic scalar potential can be modeled instead of the **H**-field directly [13, 14]. This single-output model is expressed as

$$\begin{aligned} \varphi(\mathbf{x}) &\sim \mathcal{GP}\left(0, \kappa_{\text{lin.}}(\mathbf{x}, \mathbf{x}') + \kappa_{\text{SE}}(\mathbf{x}, \mathbf{x}')\right), \\ \mathbf{y}_i &= -\nabla\varphi(\mathbf{x}_i) + \mathbf{e}_i, \quad \mathbf{e}_i \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{n}}^2\mathbf{I}_3), \end{aligned} \tag{2-15}$$

where $\kappa_{\text{lin.}}(\mathbf{x}, \mathbf{x}') = \sigma_{\text{lin.}}^2\mathbf{x}^{\top}\mathbf{x}'$ represents the constant deviations of the magnetic field components [14]. Predictions of the magnetic field are computed as the negative gradient of

the magnetic scalar potential. The predictive distribution can be derived following Särkkä's approach for Gaussian process regression models with linear operators [23], as the gradient operator is a linear operator given the linearity of differentiation. It is given by

$$\mathbb{E}\left[\mathbf{H}(\mathbf{X}_*)\right] = \partial^2(\mathbf{K}_{*,\mathbf{f}}) \left(\partial^2(\mathbf{K}_{\mathbf{f},\mathbf{f}}) + \sigma_\mathrm{n}^2\mathbf{I}\right)^{-1} \mathrm{vec}\left(\mathbf{Y}^\top\right), \tag{2-16a}$$

$$\mathbb{V}\left[\mathbf{H}(\mathbf{X}_*)\right] = \partial^2(\mathbf{K}_{*,*}) - \partial^2(\mathbf{K}_{*,\mathbf{f}}) \left(\partial^2(\mathbf{K}_{\mathbf{f},\mathbf{f}}) + \sigma_\mathrm{n}^2\mathbf{I}\right)^{-1} \partial^2(\mathbf{K}_{\mathbf{f},*}), \tag{2-16b}$$

where $\partial^2(\cdot)$ denotes the elementwise computation of the covariance matrix with the outer product of the gradient with itself, i.e., $\nabla_\mathbf{x}\kappa(\mathbf{x},\mathbf{x}')\nabla_{\mathbf{x}'}^\top$. This elementwise computation for the linear kernel leads to the 3×3 diagonal kernel function $\boldsymbol{\kappa}_{\text{const.}}(\mathbf{x},\mathbf{x}')$, describing the constant deviations of Earth's magnetic field. For the squared exponential kernel, this leads to a curl-free extension of the squared exponential kernel of equation (2-3) capable of learning curl-free vector fields [13]. This kernel function is given by

$$\boldsymbol{\kappa}_{\text{curl}}(\mathbf{x},\mathbf{x}') = \frac{\sigma_\mathrm{f}^2}{\ell^2}\exp\left(-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\ell^2}\right)\left(\mathbf{I}_3 - \left(\frac{\mathbf{x}-\mathbf{x}'}{\ell}\right)\left(\frac{\mathbf{x}-\mathbf{x}'}{\ell}\right)^\top\right). \tag{2-17}$$

Using the derived 3×3 kernel functions $\boldsymbol{\kappa}_{\text{const.}}(\mathbf{x},\mathbf{x}')$ and $\boldsymbol{\kappa}_{\text{SE}}(\mathbf{x},\mathbf{x}')$, the scalar potential model can equivalently be written as a multi-output Gaussian process model [13], given by

$$\begin{aligned} \mathbf{H}(\mathbf{x}) &\sim \mathcal{GP}\left(\mathbf{0}, \boldsymbol{\kappa}_{\text{const.}}(\mathbf{x},\mathbf{x}') + \boldsymbol{\kappa}_{\text{curl}}(\mathbf{x},\mathbf{x}')\right), \\ \mathbf{y}_i &= \mathbf{H}(\mathbf{x}_i) + \mathbf{e}_i, \quad \mathbf{e}_i \sim \mathcal{N}(\mathbf{0}, \sigma_\mathrm{n}^2\mathbf{I}_3), \end{aligned} \tag{2-18}$$

For hyperparameter learning with the scalar potential model, the scalar kernel functions in equation (2-6) are substituted by the 3×3 kernel functions $\boldsymbol{\kappa}_{\text{const.}}(\mathbf{x},\mathbf{x}')$ and $\boldsymbol{\kappa}_{\text{SE}}(\mathbf{x},\mathbf{x}')$. This leads to the log marginal likelihood function

$$\log p(\mathbf{Y}|\boldsymbol{\theta}) = \underbrace{-\frac{1}{2}\mathrm{vec}\left(\mathbf{Y}^\top\right)^\top \left(\partial^2(\mathbf{K}_{\mathbf{f},\mathbf{f}}) + \sigma_\mathrm{n}^2\mathbf{I}\right)^{-1}\mathrm{vec}\left(\mathbf{Y}^\top\right)}_{\text{data fit}} \underbrace{-\frac{1}{2}\log\left|\partial^2(\mathbf{K}_{\mathbf{f},\mathbf{f}}) + \sigma_\mathrm{n}^2\mathbf{I}\right|}_{\text{complexity penalty}} - \frac{N}{2}\log 2\pi. \tag{2-19}$$

For inference with the scalar potential model, the solution to a $3N\times 3N$ system is required. Consequently, predictions with the scalar potential model with full Gaussian process regression will already prove to become prohibitive from roughly 3,300 three-dimensional measurements. Previous research has focused on approximating the scalar potential model for magnetic field modeling, for which the Laplace operator eigenbasis approximation has been developed [14]. The approach is based on the eigendecomposition of the Laplace operator, where the eigendecomposition of the Laplace operator subject to Dirichlet boundary conditions is solved in the domain of interest [16]. The eigenfunctions and eigenvalues are used as the basis functions and the inputs of the spectral density function of the squared exponential kernel respectively, which are both used to approximate the squared exponential kernel [14] and thus Gaussian process regression. Due to the Dirichlet boundary conditions, the approximation reverts to the deviation of Earth's magnetic field near the domain boundaries with zero uncertainty. The Laplace operator eigenbasis approach reduces the computational complexity associated with Gaussian process regression to $\mathcal{O}(NM_\mathrm{bf}^2)$, where $M_\mathrm{bf} < N$ is the number of basis functions. The accuracy of the approach can be increased by increasing the number of basis functions, thus trading accuracy for efficiency. The approximation can also be integrated into the shared model [14], although this has not been done in the existing literature.

2-4 MVN Link Suit

To conduct experiments with real magnetic field measurements, data are collected using the MVN Link Suit, shown in Figure 2-1. The MVN Link suit is a motion capture suit containing 17 IMUs equipped with magnetometers distributed over the body. These IMUs provide accelerometer, gyroscope, and magnetometer data at a maximum sample rate of 240 Hz, thus providing up to 4,080 measurements per second. The magnetometers measure the magnetic field in arbitrary units, which are normalized to 1 during calibration as an absolute reference to Earth's magnetic field [24]. As the measurements are normalized and the **B**-field and the **H**-field are proportionate assuming no magnetization (equation (2-8), $\mathbf{M} = \mathbf{0}$), the measurements can be interpreted as scaled measurements of either field. Given the use of the scalar potential model, the measurements are assumed to be scaled measurements of the **H**-field. For interpretability, Xsens expects a conversion rate of approximately 0.49 µT for each arbitrary unit in the Netherlands [24], which corresponds to the World Magnetic Model [25]. Assuming no magnetization, this translates to approximately 0.39 A/m per arbitrary unit for the **H**-field.

Figure 2-1: MVN Link suit [26].

The accelerometer and gyroscope data provided by the IMUs are not directly useful for modeling magnetic field maps using Gaussian process regression, as position estimates are required and double integration of the IMU data lacks accuracy due to significant cumulative errors [6]. Fortunately, new insights may be obtained by combining the IMU data. The MVN Analyze software, also developed by Xsens, uses sensor fusion algorithms and biomechanical (human motion) models to provide supplementary data, including the position and orientation estimates of 23 segments of the body, found to be consistent for data sets spanning large amounts of time [27]. The MVN Analyze software allows for two types of processing modes: *live* and *reprocessed HD* [27, 28]. With *live* processing, the data are processed frame by frame, using the sensor fusion algorithms and biomechanical models to provide the supplementary data. The mode *reprocessed HD* enables more consistent estimates through post-processing

algorithms that require future information, such as filters and smoothers. As such, both processing modes can be used for applications where one wishes to model the magnetic field through batch estimation. If sequential estimation is desired, as is the case for SLAM algorithms, the use of the *reprocessed HD* mode is not justified, as it relies on future information that is assumed not to be available yet in a sequential scheme.

The raw IMU data and the supplementary data can be extracted from the MVN Analyze software in an MVNX file. A complete overview of the available data is presented in the MVN User Manual provided by Xsens [28], summarized in Table 2-1. Unfortunately, the position and orientation estimates are only provided for the 23 segments of the body and not for the sensors. While the MVN Analyze software provides no way of extracting the position and orientation estimates of the sensors, Xsens provided them directly using an in-house software tool. These data are provided in FT files, which can be parsed using Apache Arrow's libraries [29]. An overview of the data available in the FT files is given in Table 2-2.

Table 2-1: Overview of the available data provided by the MVN Analyze software.

Segments (23 total)	Variable space
Position	$\mathbb{R}^3$
Velocity	$\mathbb{R}^3$
Acceleration	$\mathbb{R}^3$
Orientation	$\mathbb{R}^4$
Angular velocity	$\mathbb{R}^3$
Angular acceleration	$\mathbb{R}^3$
Sensors (17 total)	**Variable space**
Free acceleration	$\mathbb{R}^3$
Magnetic field	$\mathbb{R}^3$
Orientation	$\mathbb{R}^4$
Joints (22 total)	**Variable space**
Joint angle (Euler)	$\mathbb{R}^3$
Miscellaneous	**Variable space**
Position of center of mass	$\mathbb{R}^3$
Binary foot contact data	$\{0, 1\}^4$

Table 2-2: Overview of the available data provided in the FT files.

Sensors (17 total)	Variable space
Position	$\mathbb{R}^3$
Velocity	$\mathbb{R}^3$
Acceleration	$\mathbb{R}^3$
Orientation	$\mathbb{R}^4$
Angular velocity	$\mathbb{R}^3$
Magnetic field	$\mathbb{R}^3$

Chapter 3

Scalable Magnetic Field Modeling Using Structured Kernel Interpolation

In this chapter, the integration of the SKI framework into the shared and scalar potential models is presented, answering the first research subquestion. By integrating the SKI framework the cubic computational complexity associated with modeling magnetic field maps using full Gaussian process regression is reduced while retaining sufficient accuracy. This leads to faster inference and thus allows for the use of more measurements. In Section 3-1, inducing variables are introduced that provide the required fundamentals of the SKI framework. The SKI framework itself is presented in Section 3-2, tailored to the shared and scalar potential models for magnetic field modeling that were discussed in Section 2-3. The computation of SKI-specific interpolation matrices is described in Section 3-3. Fast computation of the predictive distributions of the shared and scalar potential models using the SKI framework is discussed in Section 3-4. Lastly, the collection of magnetic field measurements for the experiments is described in Section 3-5. Hyperparameter optimization in the SKI framework is not described in this chapter as it has not been used in the experiments. For more information on hyperparameter optimization in the SKI framework, see Appendix B.

3-1 Inducing Point Methods

The SKI framework is based on inducing point methods, which are used to approximate the exact kernel function for fast computation [19]. These methods rely on the introduction of M new latent data points, the inducing points. The inducing points consist of the inducing variables $\mathbf{u} = [u_1 \quad u_2 \quad \dots \quad u_M]^\top \in \mathbb{R}^M$ at inducing inputs $\mathbf{X_u} \in \mathbb{R}^{M \times D}$ [30]. Information of all N measurements can be captured in the inducing points. The inducing points are also directly related to the test points at inputs $\mathbf{X}_* \in \mathbb{R}^{N_* \times D}$. Methods that rely on inducing points for faster inference are inspired by the Nyström approximation [31], which aims to approximate the $N \times N$ covariance matrix $\mathbf{K}_{N,N}$ through M_{rs} randomly sampled data points,

where $M_{\text{rs}} \ll N$. The approximation is given by

$$\mathbf{K}_{N,N} \approx \mathbf{K}_{N,M_{\text{rs}}} \mathbf{K}_{M_{\text{rs}},M_{\text{rs}}}^{-1} \mathbf{K}_{M_{\text{rs}},N}, \tag{3-1}$$

where $\mathbf{K}_{N,M_{\text{rs}}}$ and $\mathbf{K}_{M_{\text{rs}},N}$ are the cross-covariance matrices between the N measurement points and the M_{rs} randomly sampled data points. While the Nyström approximation can reduce the computational complexity associated with Gaussian process regression to $\mathcal{O}(NM_{\text{rs}}^2)$, a significant disadvantage of it is that it may produce negative predictive variances [15].

The Nyström approximation led to many other inducing point approaches [30], such as the common Subset of Regressors (SoR) and Fully Independent Training Conditional (FITC) approximations. The fundamental approximation for these methods is that the training and test variables, $\mathbf{f}$ and $\mathbf{f}_*$, are conditionally independent given the inducing variables $\mathbf{u}$, meaning information about the test variables $\mathbf{f}_*$ can only be derived from the training variables $\mathbf{f}$ through the inducing variables $\mathbf{u}$. The approximation of interest within the SKI framework is the Subset of Regressors (SoR) approximation [32], which assumes a deterministic relation between the training and inducing variables, and between the test and inducing variables [30]. A graphical model showing the relation between the variables for full Gaussian process regression and the SoR approximation is shown in Figure 3-1. Figure 3-1a represents full Gaussian process regression, where the training and test variables are directly related. The conditional independence between the training and test variables given the inducing variables $\mathbf{u}$ for the SoR approximation is represented by Figure 3-1b. The SoR approximation can be implemented similarly to the predictive distribution for full Gaussian process regression of equation (2-2) using a modified kernel function [30], given by

$$\kappa_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{u}) \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \kappa(\mathbf{u}, \mathbf{x}_j). \tag{3-2}$$

The modified kernel function results in new covariance matrices, which are given by

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}, \tag{3-3a}$$

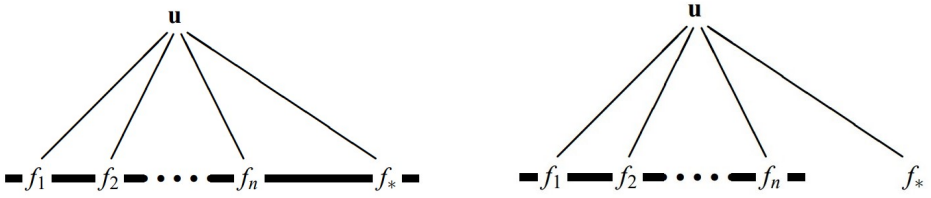
$$\mathbf{K}_{*,*} = \mathbf{K}_{*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},*}, \tag{3-3b}$$

$$\mathbf{K}_{*,\mathbf{f}} = \mathbf{K}_{*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}. \tag{3-3c}$$

The SoR approximation has an associated computational complexity of $\mathcal{O}(NM^2)$, similar to the Nyström approximation, allowing for faster inference if $M < N$. The computational complexity of the SoR approximation is dominated by multiplications involving the $N \times M$ and $N_* \times M$ cross-covariance matrices $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ and $\mathbf{K}_{*,\mathbf{u}}$ [19]. By placing the inducing variables on a Cartesian grid, efficient mathematical formulations may be exploited in the SKI framework for faster inference, as described in the next section.

3-2 Structured Kernel Interpolation for Magnetic Field Modeling

The SoR approximation can be used to efficiently model magnetic fields in the case of $M < N$. However, as this case relies on a low-rank approximation, deterioration in the accuracy of the predicted magnetic field is expected compare to full Gaussian process regression. The SKI framework aims to reduce the computational complexity associated with the SoR approximation while retaining accuracy. This requires the inducing points to be structured in a



(a) Full Gaussian process regression; the training and test variables are directly connected.

(b) SoR approximation; the training and test variables are conditionally independent.

Figure 3-1: Graphical model of the relation between the training and test function values given inducing variables $\mathbf{u}$ [30].

Cartesian grid of size $m^{(1)} \times m^{(2)} \times \cdots \times m^{(D)}$, equally spaced in every dimension, for a total of $M = \prod_{d=1}^{D} m^{(d)}$ inducing inputs. By further reducing the computational complexity, more inducing variables can be used, even $M > N$, such that more accurate estimates of the magnetic field are attained without conceding computational speed. Within the SKI framework, the cross-covariance matrices $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ and $\mathbf{K}_{*,\mathbf{u}}$ are approximated using sparse interpolation matrices, $\mathbf{W}_\mathbf{f} \in \mathbb{R}^{N \times M}$ and $\mathbf{W}_* \in \mathbb{R}^{N_* \times M}$, through the interpolation scheme

$$\mathbf{K}_{\mathbf{f},\mathbf{u}} \approx \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}}, \tag{3-4a}$$

$$\mathbf{K}_{*,\mathbf{u}} \approx \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}}. \tag{3-4b}$$

The choice of $\mathbf{W}_\mathbf{f}$ and $\mathbf{W}_*$ is further described in Section 3-3. The interpolation matrices lead to approximated versions of the SoR covariance matrices of equation (3-3), expressed as

$$\begin{aligned} \mathbf{K}_{\mathbf{f},\mathbf{f}} &= \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}} \\ &\approx \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \\ &= \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top, \end{aligned} \tag{3-5a}$$

$$\begin{aligned} \mathbf{K}_{*,*} &= \mathbf{K}_{*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},*} \\ &\approx \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_*^\top \\ &= \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_*^\top, \end{aligned} \tag{3-5b}$$

$$\begin{aligned} \mathbf{K}_{*,\mathbf{f}} &= \mathbf{K}_{*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}} \\ &\approx \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \\ &= \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top. \end{aligned} \tag{3-5c}$$

The approximated covariance matrices can be substituted into the predictive distributions for the shared and scalar potential models, given by equations (2-2) and (2-16) respectively. The entries of the interpolation matrices are only based on the training inputs, so for shared modeling the same interpolation matrices are used for each magnetic field component. The predictive distribution for shared modeling within the SKI framework is given by

$$\mathbb{E}\left[\mathbf{h}^{(d)}(\mathbf{X}_*)\right] = \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \left(\mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \mathbf{Y}^{(d)}, \tag{3-6a}$$

$$\mathbb{V}\left[\mathbf{h}^{(d)}(\mathbf{X}_*)\right] = \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_*^\top - \mathbf{W}_* \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \left(\mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_*^\top. \tag{3-6b}$$

As the predictive distribution of the scalar potential model is based on the derivatives of the magnetic scalar potential, the elementwise computation $\partial^2(\cdot)$ of the covariance matrices can be approximated using SKI with derivatives (D-SKI) [33]. Using D-SKI, $\partial^2(\cdot)$ can be simplified through differentiation of the interpolation scheme, given by

$$\mathbf{K}_{\mathbf{f},\mathbf{u}} \approx (\partial \mathbf{W}_{\mathbf{f}})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}, \tag{3-7a}$$

$$\mathbf{K}_{*,\mathbf{u}} \approx (\partial \mathbf{W}_{*})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}, \tag{3-7b}$$

where $\partial \mathbf{W}_{\mathbf{f}} \in \mathbb{R}^{3N \times M}$ and $\partial \mathbf{W}_{*} \in \mathbb{R}^{3N_* \times M}$, due to the three components of the magnetic field. The computation of the differentiated interpolation matrices is described in Section 3-3. The predictive distribution of the scalar potential model for magnetic field modeling using D-SKI is given by

$$\mathbb{E}\left[\mathbf{H}(\mathbf{X}_*)\right] = (\partial \mathbf{W}_{*})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}\, (\partial \mathbf{W}_{\mathbf{f}})^\top \left((\partial \mathbf{W}_{\mathbf{f}})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}\, (\partial \mathbf{W}_{\mathbf{f}})^\top + \sigma_{\mathrm{n}}^2 \mathbf{I}\right)^{-1} \mathrm{vec}\left(\mathbf{Y}^\top\right), \tag{3-8a}$$

$$\begin{aligned}\mathbb{V}\left[\mathbf{H}(\mathbf{X}_*)\right] = {} & (\partial \mathbf{W}_{*})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}\, (\partial \mathbf{W}_{*})^\top - \\ & (\partial \mathbf{W}_{*})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}\, (\partial \mathbf{W}_{\mathbf{f}})^\top \left((\partial \mathbf{W}_{\mathbf{f}})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}\, (\partial \mathbf{W}_{\mathbf{f}})^\top + \sigma_{\mathrm{n}}^2 \mathbf{I}\right)^{-1} (\partial \mathbf{W}_{\mathbf{f}})\, \mathbf{K}_{\mathbf{u},\mathbf{u}}\, (\partial \mathbf{W}_{*})^\top .\end{aligned} \tag{3-8b}$$

To have the (D-)SKI predictive distributions accurately represent the full predictive distributions, the (differentiated) sparse interpolation matrices must lead to good approximations of the covariance matrices. Depending on the interpolation method used, the number of non-zero entries per row of $\mathbf{W}_{\mathbf{f}}$ and $\mathbf{W}_{*}$ changes and thus the sparsity. The developers of the (D-)SKI framework suggest using convolution interpolation algorithms for good accuracy with retained computational speed [19, 33]. The computation of the sparse interpolation matrices using convolution interpolation algorithms is discussed in the next section.

3-3 Sparse Convolution Interpolation Matrices

For accurate estimation of the magnetic field using the SKI framework, proper sparse interpolation matrices $\mathbf{W}_{\mathbf{f}}$ and $\mathbf{W}_{*}$ are needed. Computation of their differentiated alternatives $\partial \mathbf{W}_{\mathbf{f}}$ and $\partial \mathbf{W}_{*}$ follows from the definition of $\mathbf{W}_{\mathbf{f}}$ and $\mathbf{W}_{*}$. The interpolation matrices $\mathbf{W}_{\mathbf{f}}$ and $\mathbf{W}_{*}$ consist of interpolation weights for each training or test input with respect to the inducing inputs $\mathbf{X}_{\mathbf{u}}$. As such, $\mathbf{W}_{\mathbf{f}}$ and $\mathbf{W}_{*}$ consist of N and N_* row-concatenated row vectors $\mathbf{w}_i$ and $\mathbf{w}_{*,i}$ respectively, i.e.,

$$\mathbf{W}_{\mathbf{f}} = \begin{bmatrix} \mathbf{w}_1(\mathbf{x}_1) \\ \mathbf{w}_2(\mathbf{x}_2) \\ \vdots \\ \mathbf{w}_N(\mathbf{x}_N) \end{bmatrix}, \quad \mathbf{W}_{*} = \begin{bmatrix} \mathbf{w}_{*,1}(\mathbf{x}_{*,1}) \\ \mathbf{w}_{*,2}(\mathbf{x}_{*,2}) \\ \vdots \\ \mathbf{w}_{*,N_*}(\mathbf{x}_{*,N_*}) \end{bmatrix}. \tag{3-9}$$

The number of interpolation weights for each input depends on the dimensionality of the inputs, D, and the interpolation method used. As the inducing inputs are structured on a Cartesian grid, equispaced in each dimension, the interpolation weights can be computed as a Kronecker product over the D input dimensions [34], i.e.,

$$\mathbf{w}_i(\mathbf{x}_i) = \bigotimes_{d=1}^{D} \mathbf{w}_i^{(d)}\left(x_i^{(d)}\right), \quad \mathbf{w}_i \in \mathbb{R}^{1 \times m^{(d)}}, \quad i = 1, 2, \ldots, N. \tag{3-10}$$

The interpolation weights can thus be computed separately for every dimension of the input space, requiring independent computation on each component $x^{(d)}$ of the training or test input $\mathbf{x} \in \mathbb{R}^D$. The differentiated interpolation matrices $\partial\mathbf{W_f} \in \mathbb{R}^{DN \times M}$ and $\partial\mathbf{W}_* \in \mathbb{R}^{DN_* \times M}$ consist of the row-concatenated gradients of the row vectors $\mathbf{w}_i$, i.e.,

$$\partial\mathbf{W_f} = \begin{bmatrix} \nabla\mathbf{w}_1(\mathbf{x}_1) \\ \nabla\mathbf{w}_2(\mathbf{x}_2) \\ \vdots \\ \nabla\mathbf{w}_N(\mathbf{x}_N) \end{bmatrix}, \quad \partial\mathbf{W}_* = \begin{bmatrix} \nabla\mathbf{w}_{*,1}(\mathbf{x}_{*,1}) \\ \nabla\mathbf{w}_{*,2}(\mathbf{x}_{*,2}) \\ \vdots \\ \nabla\mathbf{w}_{*,N_*}(\mathbf{x}_{*,N_*}) \end{bmatrix}. \tag{3-11}$$

As the interpolation weights are computed as a Kronecker product over the dimensions, the gradient in a specific dimension can be computed as the derivative of the interpolation weights in that direction, Kronecker multiplied with the regular interpolation weights in the other directions. Given the SKI framework is used to approximate the predictive distributions for magnetic field modeling in a three-dimensional input space ($D = 3$), the gradient can be expressed as

$$\begin{aligned} \frac{\partial\mathbf{w}(\mathbf{x})}{\partial x^{(1)}} &= \frac{\partial\mathbf{w}^{(1)}(x^{(1)})}{\partial x^{(1)}} \otimes \mathbf{w}^{(2)}\left(x^{(2)}\right) \otimes \mathbf{w}^{(3)}\left(x^{(3)}\right), \\ \frac{\partial\mathbf{w}(\mathbf{x})}{\partial x^{(2)}} &= \mathbf{w}^{(1)}\left(x^{(1)}\right) \otimes \frac{\partial\mathbf{w}^{(2)}(x^{(2)})}{\partial x^{(2)}} \otimes \mathbf{w}^{(3)}\left(x^{(3)}\right), \\ \frac{\partial\mathbf{w}(\mathbf{x})}{\partial x^{(3)}} &= \mathbf{w}^{(1)}\left(x^{(1)}\right) \otimes \mathbf{w}^{(2)}\left(x^{(2)}\right) \otimes \frac{\partial\mathbf{w}^{(3)}(x^{(3)})}{\partial x^{(3)}}. \end{aligned} \tag{3-12}$$

This formulation is generalizable to any arbitrary dimensionality D.

To efficiently estimate the magnetic field while retaining accuracy, convolution interpolation algorithms are used to compute the sparse interpolation matrices and their derivatives. These algorithms use an interpolation kernel to achieve accurate interpolation, which ideally is given by the sinc-function, following the Nyquist-Shannon sampling theorem (see the Whittaker-Shannon interpolation formula [35]). This ideal interpolation kernel cannot be used in practice, however, so nth-order polynomial kernels resembling the sinc-function are used as a strong alternative [36]. The developers of SKI suggest the use of cubic (third order) convolution interpolation for good accuracy while retaining computational speed [19]. Cubic convolution interpolation requires up to four interpolation weights per dimension d for each training or test input depending on the proximity to the boundaries [37]. The total number of interpolation weights per training or test input scales as 4^D, which is easily observed through the Kronecker product of equation (3-10). For higher accuracy, the developers of D-SKI suggest the use of quintic (fifth order) convolution interpolation [33]. Quintic convolution interpolation requires six interpolation weights per dimension d, resulting in a total number of 6^D weights per training or test input, thus requiring more operations per matrix operation. For magnetic field modeling with a three-dimensional input space, matrix multiplications with quintic convolution interpolation matrices require 216 elementwise computations per training or test input, whereas cubic convolution interpolation matrices require up to 64 elementwise computations (may be less due to the boundary conditions).

3-3-1 Convolution Interpolation

Considering the rows of the sparse interpolation matrices $\mathbf{w}_i$ are computed through the Kronecker product over $\mathbf{w}_i^{(d)} \in \mathbb{R}^{1 \times m^{(d)}}$ for all d, only the one-dimensional convolution interpo-

lation algorithms are required. The algorithms rely on the computation of the interpolation weights for one dimension of a training or test input, $x_i^{(d)}$ of $\mathbf{x}_i \in \mathbb{R}^D$, based on its location $x_i^{(d)}$ with respect to the locations of the inducing points in that dimension, $\mathbf{X}_\mathbf{u}^{(d)} \in \mathbb{R}^{m^{(d)}}$. To determine the interpolation weights, the assumption is made that $x_{u,k}^{(d)}$ is the inducing point of $\mathbf{X}_\mathbf{u}^{(d)} = [x_{u,1}^{(d)} \quad x_{u,2}^{(d)} \quad \dots \quad x_{u,m^{(d)}}^{(d)}]^\top \in \mathbb{R}^{m^{(d)}}$ prior to the training or test input $x_i^{(d)}$ in dimension d, i.e.,

$$x_{u,k}^{(d)} \leq x_i^{(d)} < x_{u,k+1}^{(d)}. \tag{3-13}$$

The interpolation function for nth-order convolution interpolation with equally spaced inducing points can be written as a weighted sum of the nth-order interpolation kernel $\gamma_n(\cdot)$ [37], i.e.,

$$g_n\left(x_i^{(d)}\right) = \sum_k c_k \gamma_n\left(\frac{x_i^{(d)} - x_{u,k}^{(d)}}{\delta^{(d)}}\right), \tag{3-14}$$

where $\delta^{(d)}$ is the distance between the equispaced inducing points in dimension d. The kernel coefficients c_k do not need to be computed explicitly as they are the values of the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$. The interpolation weights are multiplied with $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ in the interpolation scheme of equation (3-4), thus fitting the interpolation function of equation (3-14). The interpolation kernel is computed as $\gamma_n(s)$, where the interpolation variable s is defined as

$$s = \frac{x^{(d)} - x_{u,k}^{(d)}}{\delta^{(d)}}. \tag{3-15}$$

Considering equations (3-13) and (3-15), the relation $0 \leq s < 1$ holds. The interpolation kernels for cubic and quintic convolution interpolation are respectively given by equations (3-16a) and (3-16b) [36]. These functions are third- and fifth-order polynomial kernels resembling the sinc-function, which are required to be continuous, equal to one at $s = 0$, and zero at other integer values of s [37].

$$\gamma_3(s) = \begin{cases} \frac{3}{2}|s|^3 - \frac{5}{2}|s|^2 + 1 & 0 \leq |s| < 1, \\ -\frac{1}{2}|s|^3 + \frac{5}{2}|s|^2 - 4|s| + 2 & 1 \leq |s| < 2, \\ 0 & 2 \leq |s|. \end{cases} \tag{3-16a}$$

$$\gamma_5(s) = \begin{cases} -\frac{27}{32}|s|^5 + \frac{63}{32}|s|^4 - \frac{17}{8}|s|^2 + 1 & 0 \leq |s| < 1, \\ \frac{13}{64}|s|^5 - \frac{21}{16}|s|^4 + \frac{85}{32}|s|^3 - \frac{7}{8}|s|^2 - \frac{165}{64}|s| + \frac{61}{32} & 1 \leq |s| < 2, \\ \frac{3}{64}|s|^5 - \frac{21}{32}|s|^4 + \frac{117}{32}|s|^3 - \frac{81}{8}|s|^2 + \frac{891}{64}|s| - \frac{243}{32} & 2 \leq |s| < 3, \\ 0 & 3 \leq |s|. \end{cases} \tag{3-16b}$$

For the computation of the derivative matrices $\partial\mathbf{W}_\mathbf{f}$ and $\partial\mathbf{W}_*$ for the scalar potential model, the derivatives of the interpolation kernels are additionally needed. Both interpolation kernels have continuous first derivatives, which can be derived using the chain rule and are respectively given by

$$\frac{\partial\gamma_3(s)}{\partial x^{(d)}} = \frac{1}{\delta^{(d)}} \begin{cases} \frac{9}{2}|s|s - 5s & 0 \leq |s| < 1, \\ -\frac{3}{2}|s|s + 5s - 4\frac{s}{|s|} & 1 \leq |s| < 2, \\ 0 & 2 \leq |s|, \end{cases} \tag{3-17a}$$

$$\frac{\partial \gamma_5(s)}{\partial x^{(d)}} = \frac{1}{\delta^{(d)}} \begin{cases} -\frac{135}{32}|s|^3 s + \frac{63}{32}|s|^2 s - \frac{17}{4}s & 0 \leq |s| < 1, \\ \frac{65}{64}|s|^3 s - \frac{21}{4}s^3 + \frac{255}{32}|s|s - \frac{7}{4}s - \frac{165}{64}\frac{s}{|s|} & 1 \leq |s| < 2, \\ \frac{15}{64}|s|^3 s - \frac{21}{8}s^3 + \frac{351}{32}|s|s - \frac{81}{4}s + \frac{891}{64}\frac{s}{|s|} & 2 \leq |s| < 3, \\ 0 & 3 \leq |s|. \end{cases} \tag{3-17b}$$

Considering $\gamma_3(s) = 0$ for $|s| \geq 2$, only the coefficients c_{k-1}, c_k, c_{k+1} and c_{k+2} are needed for input $x^{(d)}$ with cubic convolution interpolation. For quintic convolution interpolation, the coefficients c_{k-2} and c_{k+3} are additionally needed. Following the relation $x^{(d)} - x^{(d)}_{u,k-1} = (x^{(d)} - x^{(d)}_{u,k}) + (x^{(d)}_{u,k} - x^{(d)}_{u,k-1}) = \delta^{(d)}s + \delta^{(d)}$, the interpolation functions for cubic and quintic convolution interpolation are given by

$$g_3(s) = c_{k-1}\gamma_3(s+1) + c_k\gamma_3(s) + c_{k+1}\gamma_3(s-1) + c_{k+2}\gamma_3(s-2), \tag{3-18a}$$

$$\begin{aligned} g_5(s) = c_{k-2}\gamma_5(s+2) + c_{k-1}\gamma_5(s+1) + c_k\gamma_5(s) + c_{k+1}\gamma_5(s-1) + \\ c_{k+2}\gamma_5(s-2) + c_{k+3}\gamma_5(s-3). \end{aligned} \tag{3-18b}$$

An advantage of cubic over quintic convolution interpolation is that it has well-defined boundary conditions [37]. The boundary conditions are given by

$$c_0 = 3c_1 - 3c_2 + c_3, \tag{3-19a}$$

$$c_{m^{(d)}+1} = 3c_{m^{(d)}} - 3c_{m^{(d)}-1} + c_{m^{(d)}-2}. \tag{3-19b}$$

These boundary conditions allow the chosen inducing input grid to closely contain the training and test inputs of interest, only requiring one inducing point outside the training and test inputs on each side. For quintic convolution interpolation, two additional inducing inputs are needed, thus covering a larger D-dimensional space when using the same number of inducing inputs, resulting in a loss of accuracy. The impact of both cubic and quintic convolution interpolation is studied using simulations and the experimental setup in Chapter 4.

3-3-2 Constructing the Sparse Interpolation Matrices

Construction of the sparse interpolation matrices $\mathbf{W_f}$ and $\mathbf{W}_*$ for the shared model is described by equation (3-9). For the scalar potential model, computation of $\partial\mathbf{W_f}$ and $\partial\mathbf{W}_*$ is described by equation (3-11). Computation of the interpolation weights vectors $\mathbf{w}_i$ and their derivatives is described by Tables 3-1 and 3-2 for cubic and quintic convolution interpolation respectively. The interpolation weights $\gamma_3(\cdot)$ and $\gamma_5(\cdot)$ are inserted into the interpolation vector $\mathbf{w}_i^{(d)}$ at their corresponding indices. The interpolation weights for cubic convolution interpolation include the boundary conditions, whereas the quintic convolution interpolation weights are constrained to the interval $3 \leq k \leq m^{(d)} - 3$. For D-SKI, the derivative interpolation weights are inserted into the derivative interpolation weights vectors at the same indices as the regular interpolation weights. As mentioned previously, the gradient of $\mathbf{w}_i$ in a specific dimension can be computed as the Kronecker product of the derivative interpolation weights in that dimension with the regular interpolation weights in the other dimension (equation (3-12)).

Computation of $\mathbf{W_f}$ and $\mathbf{W}_*$ for the shared model is described by Algorithm 1. The interpolation matrices $\partial\mathbf{W_f}$ and $\partial\mathbf{W}_*$ for the scalar potential model can be computed following

Algorithm 2. The sparse interpolation matrices, combined with the grid-specific structure in $\mathbf{K}_{\mathbf{u},\mathbf{u}}$, allow for fast computation of the predictive distributions of the shared and scalar potential models of equations (3-6) and (3-8) through efficient formulations as described in the next section.

Table 3-1: Overview of where to insert the cubic convolution interpolation weights into $\mathbf{w}_i^{(d)}$.

Condition	Weights	Indices
$k = 1$	$\begin{bmatrix} \gamma_3(s) + 3\gamma_3(s+1) \\ \gamma_3(s-1) - 3\gamma_3(s+1) \\ \gamma_3(s-2) + \gamma_3(s+1) \end{bmatrix}$	$\begin{Bmatrix} k \\ k+1 \\ k+2 \end{Bmatrix}$
$2 \leq k \leq m^{(d)} - 2$	$\begin{bmatrix} \gamma_3(s+1) \\ \gamma_3(s) \\ \gamma_3(s-1) \\ \gamma_3(s-2) \end{bmatrix}$	$\begin{Bmatrix} k-1 \\ k \\ k+1 \\ k+2 \end{Bmatrix}$
$k = m^{(d)} - 1$	$\begin{bmatrix} \gamma_3(s+1) + \gamma_3(s-2) \\ \gamma_3(s) - 3\gamma_3(s-2) \\ \gamma_3(s-1) + 3\gamma_3(s-2) \end{bmatrix}$	$\begin{Bmatrix} k-1 \\ k \\ k+1 \end{Bmatrix}$
$k = m_d$	$\begin{bmatrix} \gamma_3(s) \end{bmatrix}$	$\begin{Bmatrix} k \end{Bmatrix}$

Table 3-2: Overview of where to insert the quintic convolution interpolation weights into $\mathbf{w}_i^{(d)}$.

Condition	Weights	Indices
$3 \leq k \leq m^{(d)} - 3$	$\begin{bmatrix} \gamma_5(s+2) \\ \gamma_5(s+1) \\ \gamma_5(s) \\ \gamma_5(s-1) \\ \gamma_5(s-2) \\ \gamma_5(s-3) \end{bmatrix}$	$\begin{Bmatrix} k-2 \\ k-1 \\ k \\ k+1 \\ k+2 \\ k+3 \end{Bmatrix}$

Algorithm 1 Computation of $\mathbf{W}$ with input dimensionality D

Input: $\mathbf{X}^{(d)} \in \mathbb{R}^N$, $\mathbf{X}_\mathbf{u}^{(d)} \in \mathbb{R}^{m^{(d)}}$ for $d = 1$ to D
Output: $\mathbf{W} \in \mathbb{R}^{N \times M}$

for $i = 1$ to N **do**
  for $d = 1$ to D **do**
    define $x_i^{(d)}$ as the ith value of $\mathbf{X}^{(d)}$
    find $x_{u,k}^{(d)}$ in $\mathbf{X}_\mathbf{u}^{(d)}$ nearest $x_i^{(d)}$ that satisfies $x_{u,k}^{(d)} \leq x_i^{(d)}$
    compute s
    compute $\gamma(\cdot)$ using the cubic or quintic interpolation kernel (equation (3-16))
    insert $\gamma(\cdot)$ into $\mathbf{w}_i^{(d)}$ as described by either Table 3-1 or Table 3-2
  end for
  compute $\mathbf{w}_i$ using the Kronecker product (equation (3-10))
end for
row-concatenate all $\mathbf{w}_i$ to find $\mathbf{W}$ (equation (3-9))

Algorithm 2 Computation of $\partial\mathbf{W}$ with input dimensionality D

Input: $\mathbf{X}^{(d)} \in \mathbb{R}^N$, $\mathbf{X}_\mathbf{u}^{(d)} \in \mathbb{R}^{m^{(d)}}$ for $d = 1$ to D
Output: $\partial\mathbf{W} \in \mathbb{R}^{DN \times M}$

for $i = 1$ to N **do**
  for $d = 1$ to D **do**
    define $x_i^{(d)}$ as the ith value of $\mathbf{X}^{(d)}$
    find $x_{u,k}^{(d)}$ in $\mathbf{X}_\mathbf{u}^{(d)}$ nearest $x_i^{(d)}$ that satisfies $x_{u,k}^{(d)} \leq x_i^{(d)}$
    compute s
    compute the cubic or quintic interpolation kernel $\gamma(\cdot)$ (equation (3-16))
    insert $\gamma(\cdot)$ into $\mathbf{w}_i^{(d)}(x_i^{(d)})$ as described by either Table 3-1 or Table 3-2
    compute the partial derivatives of $\gamma(\cdot)$ (equation (3-17))
    insert derivatives into $\partial(\mathbf{w}_i^{(d)}(x_i^{(d)}))/\partial(x^{(d)})$ as described by Table 3-1 or Table 3-2
  end for
  compute $\nabla\mathbf{w}_i(\mathbf{x}_i)$ using D partial derivative Kronecker products (equation (3-12))
end for
row-concatenate all $\nabla\mathbf{w}_i(\mathbf{x}_i)$ to find $\partial\mathbf{W}$ (equation (3-11))

3-4 Scalable Magnetic Field Modeling

Following the computation of the sparse interpolation matrices using convolution interpolation in the previous section, the predictive distributions of the shared and scalar potential models (equations (3-6) and (3-8)) can be computed using efficient mathematical formulations. The formulations used in the SKI framework are part of a family called Krylov subspace methods, which are used to find approximate solutions to large linear systems [38]. Krylov subspace methods make use of the Krylov subspace of a square $N \times N$ matrix $\mathbf{A}$ and an arbitrary vector $\mathbf{v} \in \mathbb{R}^N$. The definition of the arbitrary vector is dependent on the Krylov subspace method. The Krylov subspace for square matrix $\mathbf{A}$ and vector $\mathbf{v}$ is given by

$$\mathcal{K}(\mathbf{A}, \mathbf{v}) = \text{span}\left\{\mathbf{v}, \mathbf{A}\mathbf{v}, \ldots, \mathbf{A}^{N-1}\mathbf{v}\right\}. \tag{3-20}$$

The potential of Krylov subspace methods is easily observed through the structure of the subspace, where each vector spanning the subspace is iteratively found through matrix-vector multiplication (MVM) of $\mathbf{A}$ with the previous vector, starting with probe vector $\mathbf{v}$. This avoids matrix-matrix multiplications involving the powers of $\mathbf{A}$. Within the SKI framework, Krylov subspace methods are used to efficiently solve systems involving the matrix inverse. To simplify notation, the matrices $\mathbf{A}$ and $\partial\mathbf{A}$ are defined as the matrix to be inverted in the predictive distributions of the shared and scalar potential models respectively. Additionally, the matrices $\mathbf{C}$ and $\partial\mathbf{C}$ are defined as the precomputable parts of the predictive variances of the shared and scalar potential models, which are only based on training points (not test points). The matrices $\mathbf{A}$ and $\mathbf{C}$ are expressed within the predictive distribution of the shared model (equation (3-6)) as

$$\mathbb{E}\left[\mathbf{h}^{(d)}(\mathbf{X}_*)\right] = \mathbf{W}_*\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_{\mathbf{f}}^\top \underbrace{\left(\mathbf{W}_{\mathbf{f}}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_{\mathbf{f}}^\top + \sigma_{\text{n}}^2\mathbf{I}\right)}_{\mathbf{A}}{}^{-1}\mathbf{Y}^{(d)}, \tag{3-21a}$$

$$\mathbb{V}\left[\mathbf{h}^{(d)}(\mathbf{X}_*)\right] = \mathbf{W}_*\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_*^\top - \mathbf{W}_*\underbrace{\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_{\mathbf{f}}^\top\left(\mathbf{W}_{\mathbf{f}}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_{\mathbf{f}}^\top + \sigma_{\text{n}}^2\mathbf{I}\right)^{-1}\mathbf{W}_{\mathbf{f}}\mathbf{K}_{\mathbf{u},\mathbf{u}}}_{\mathbf{C}}\mathbf{W}_*^\top. \tag{3-21b}$$

The matrices $\partial\mathbf{A}$ and $\partial\mathbf{C}$ are expressed for the scalar potential model within its corresponding predictive distribution (equation (3-8)) as

$$\mathbb{E}\left[\mathbf{H}(\mathbf{X}_*)\right] = (\partial\mathbf{W}_*)\,\mathbf{K}_{\mathbf{u},\mathbf{u}}\,(\partial\mathbf{W}_{\mathbf{f}})^\top \underbrace{\left((\partial\mathbf{W}_{\mathbf{f}})\,\mathbf{K}_{\mathbf{u},\mathbf{u}}\,(\partial\mathbf{W}_{\mathbf{f}})^\top + \sigma_{\text{n}}^2\mathbf{I}\right)}_{\partial\mathbf{A}}{}^{-1}\text{vec}(\mathbf{Y}^\top), \tag{3-22a}$$

$$\begin{aligned}\mathbb{V}\left[\mathbf{H}(\mathbf{X}_*)\right] = {} & (\partial\mathbf{W}_*)\,\mathbf{K}_{\mathbf{u},\mathbf{u}}\,(\partial\mathbf{W}_*)^\top - \\ & (\partial\mathbf{W}_*)\underbrace{\mathbf{K}_{\mathbf{u},\mathbf{u}}\,(\partial\mathbf{W}_{\mathbf{f}})^\top\left((\partial\mathbf{W}_{\mathbf{f}})\,\mathbf{K}_{\mathbf{u},\mathbf{u}}\,(\partial\mathbf{W}_{\mathbf{f}})^\top + \sigma_{\text{n}}^2\mathbf{I}\right)^{-1}(\partial\mathbf{W}_{\mathbf{f}})\,\mathbf{K}_{\mathbf{u},\mathbf{u}}}_{\partial\mathbf{C}}(\partial\mathbf{W}_*)^\top.\end{aligned} \tag{3-22b}$$

Both predictive distributions rely on solutions of large systems, involving the matrices $\mathbf{A}$ and $\partial\mathbf{A}$ respectively. In numerical implementations, the solutions to these systems are often computed using numerically stable decompositions, such as the Cholesky decomposition or

the QR decomposition [38]. Certain Krylov subspace methods can be used to iteratively find the solutions to these problems through MVMs with the matrices $\mathbf{A}$ and $\partial\mathbf{A}$. The desire to use Krylov subspace methods becomes apparent when analyzing the structure of $\mathbf{A}$ and $\partial\mathbf{A}$. MVMs involving the matrix $\mathbf{A}$ can be expressed as

$$\begin{aligned}\mathbf{A}\mathbf{v} &= \left(\mathbf{W}_\mathbf{f}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2\mathbf{I}\right)\mathbf{v} \\ &= \mathbf{W}_\mathbf{f}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_\mathbf{f}^\top\mathbf{v} + \sigma_\mathrm{n}^2\mathbf{v}.\end{aligned} \tag{3-23}$$

Any equation involving $\partial\mathbf{A}$ can be similarly expressed, simply substituting $\partial\mathbf{W}_\mathbf{f}$ for $\mathbf{W}_\mathbf{f}$. As the interpolation matrices $\mathbf{W}_\mathbf{f}$ and $\partial\mathbf{W}_\mathbf{f}$ are very sparse, only containing 4^D or 6^D non-zero entries per row with cubic and quintic convolution interpolation respectively, MVMs involving these matrices can be computed very efficiently. Additionally, the grid-specific structure in the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ may be used to further speed up MVMs involving $\mathbf{A}$ or $\partial\mathbf{A}$. This is further discussed in Section 3-4-1.

For fast estimation of the magnetic field in the SKI framework, the predictive means of equations (3-21a) and (3-22a) can be computed efficiently using the conjugate gradient method [10]. In this Krylov subspace approach, MVMs are used to iteratively find the solution $\boldsymbol{\alpha}$ to the linear system

$$\underbrace{\left(\mathbf{W}_\mathbf{f}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2\mathbf{I}\right)}_{\mathbf{A}}\boldsymbol{\alpha} = \mathbf{Y}^{(d)}, \tag{3-24}$$

or the linear system $(\partial\mathbf{A})\boldsymbol{\alpha} = \mathrm{vec}(\mathbf{Y}^\top)$ for the scalar potential model. Further elaboration on the conjugate gradient method is given in Section 3-4-2, including preconditioning for faster computation. This approach can be extended for the predictive variances of equations (3-21b) and (3-22b), computing $\mathbf{C}$ and $\partial\mathbf{C}$ by solving the systems $\mathbf{A}^{-1}\left(\mathbf{W}_\mathbf{f}\mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$ and $\partial\mathbf{C}((\partial\mathbf{W}_\mathbf{f})\,\mathbf{K}_{\mathbf{u},\mathbf{u}})$ for each column of $\mathbf{W}_\mathbf{f}\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and $(\partial\mathbf{W}_\mathbf{f})\,\mathbf{K}_{\mathbf{u},\mathbf{u}}$ sequentially. However, $\mathbf{C}$ and $\partial\mathbf{C}$ can be estimated in a much more efficient manner using the Lanczos tridiagonalization algorithm [20]. This algorithm is also a Krylov subspace method, only requiring MVMs with the system matrices $\mathbf{A}$ and $\partial\mathbf{A}$. Efficient computation of the predictive variances using the Lanczos tridiagonalization algorithm is discussed in Section 3-4-3. Fast test-time predictions of the magnetic field using the results of the preconditioned conjugate gradient method and the Lanczos tridiagonalization algorithm is described in Section 3-4-4.

3-4-1 Fast Kronecker Matrix-Vector Multiplication

The grid-specific structure in the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ can be used to speed up magnetic field estimation by efficiently computing MVMs involving the matrix $\mathbf{A}$ or $\partial\mathbf{A}$ described by equation (3-23). MVMs involving $\mathbf{A}$ or $\partial\mathbf{A}$ consist of two separate terms, of which the first term can be computed in three separate MVMs. The first MVM is efficiently computed in the SKI framework as $\mathbf{W}_\mathbf{f}^\top\mathbf{v} = \mathbf{v}_\mathbf{f} \in \mathbb{R}^M$, exploiting the sparsity of $\mathbf{W}_\mathbf{f} \in \mathbb{R}^{N\times M}$. As $\mathbf{v}_\mathbf{f} \in \mathbb{R}^M$ is a vector, $\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{v}_\mathbf{f}$ is a dense MVM, which has a computational complexity of $\mathcal{O}(M^2)$. However, the grid-specific structure in $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ may lead to a reduced MVM computational complexity. For magnetic field modeling, the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is computed on a D-dimensional Cartesian grid equispaced in each dimension using the composite kernel functions $\kappa_{\text{const.}}(\mathbf{x},\mathbf{x}') + \kappa_{\text{SE}}(\mathbf{x},\mathbf{x}')$ for the shared model (equation (2-13)) and $\kappa_{\text{lin.}}(\mathbf{x},\mathbf{x}') + \kappa_{\text{SE}}(\mathbf{x},\mathbf{x}')$ for the scalar potential model (equation (2-15)). As the inducing inputs are structured in a

Cartesian grid and the squared exponential and constant kernels (equations (2-3) and (2-4)) decompose as a product kernel over the grid dimensions, their covariance matrices can be expressed as Kronecker products [19, 39], i.e.,

$$\mathbf{K}_{\mathbf{u},\mathbf{u}}^{\text{SE}} = \bigotimes_{d=1}^{D} \left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{\text{SE}}\right)^{(d)}, \tag{3-25a}$$

$$\mathbf{K}_{\mathbf{u},\mathbf{u}}^{\text{const.}} = \bigotimes_{d=1}^{D} \left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{\text{const.}}\right)^{(d)}, \tag{3-25b}$$

where the covariance matrices per dimension are scaled versions of the kernels over scalar inputs with amplitudes $\sigma_{\text{f}}^{2/D}$ and $\sigma_{\text{c}}^{2/D}$ [39]. Unfortunately, the linear kernel cannot be decomposed as a product kernel as it is a dot product kernel. Dot product kernels do not result in any favorable structure in the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{\text{lin.}}$ [39, 40], so the linear kernel cannot be used for fast inference with the scalar potential model involving exploitation of the structure of $\mathbf{K}_{\mathbf{u},\mathbf{u}}$. As mentioned in Section 2-1, non-zero means in the training data may also be handled by subtracting the mean from the training data and using the shifted measurements, although this loses the uncertainty estimation of the constant deviation of Earth's magnetic field. As the linear kernel cannot be used within the SKI framework involving fast MVMs, only the squared exponential kernel is considered for both the shared and scalar potential models from here on for consistency. Fast MVMs with the shared model using an added constant kernel are still feasible, however, which are described in Appendix A.

The Kronecker structure present in $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ constructed with the squared exponential kernel can be exploited to reduce the load of computing $\mathbf{W}_{\mathbf{f}}\mathbf{K}_{\mathbf{u},\mathbf{u}}(\mathbf{W}_{\mathbf{f}}^{\top}\mathbf{v})$. The MVM can be expressed using the Kronecker product as

$$\mathbf{W}_{\mathbf{f}}\mathbf{K}_{\mathbf{u},\mathbf{u}}\left(\mathbf{W}_{\mathbf{f}}^{\top}\mathbf{v}\right) = \mathbf{W}_{\mathbf{f}}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{v}_{\mathbf{f}} = \mathbf{W}_{\mathbf{f}}\left(\bigotimes_{d=1}^{D}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}\right)\mathbf{v}_{\mathbf{f}} \tag{3-26}$$

Computation of the Kronecker MVM for efficient implementation is described by Algorithm 3 [41]. The algorithm iteratively solves the equation

$$\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{v}_{\mathbf{f}} = \left(\bigotimes_{d=1}^{D}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}\right)\mathbf{v}_{\mathbf{f}} = \text{vec}\left(\left[\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(1)}, \ldots \left[\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(D-1)}, \left[\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(D)}, \mathbf{V}^{(D)}\right]\right]\right]\right), \tag{3-27a}$$

where the bracket notation is defined as

$$\left[\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}, \mathbf{V}^{(d)}\right] = \text{reshape}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}\mathbf{V}^{(d)}\right)^{\top}, m^{(d)}, \frac{M}{m^{(d)}}\right), \tag{3-27b}$$

with initial $\mathbf{V}^{(D)}$ defined as

$$\mathbf{V}^{(D)} = \text{reshape}\left(\mathbf{v}_{\mathbf{f}}, m^{(D)}, \frac{M}{m^{(D)}}\right). \tag{3-27c}$$

The MVM $\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{v}_{\mathbf{f}}$ is computed with a complexity of $\mathcal{O}(DM^{1+1/D})$ for grids with an equal number of inducing points per dimension [41], which is generalizable to $\mathcal{O}(M\sum_{d=1}^{D}m^{(d)})$ for arbitrarily sized grids. This computational complexity is much lower than the $\mathcal{O}(M^2)$ computational complexity of standard dense MVM with full covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}} \in \mathbb{R}^{M\times M}$. The remaining MVM with $\mathbf{W}_{\mathbf{f}}$ is again very efficient due to the sparsity of $\mathbf{W}_{\mathbf{f}}$.

Algorithm 3 Kronecker matrix-vector multiplication (MVM)

Input: $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)} \in \mathbb{R}^{m^{(d)} \times m^{(d)}}$ for $d = 1$ to D, $\mathbf{v_f} \in \mathbb{R}^M$
Output: $\mathbf{K_{u,u}v_f} \in \mathbb{R}^M$
1: $\mathbf{V}^{(D)} = \text{reshape}\left(\mathbf{v_f}, m^{(D)}, \frac{M}{m^{(d)}}\right)$
2: **for** $d = D$ to 1 **do**
3: compute $\mathbf{V}^{(d-1)} = \text{reshape}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}\mathbf{V}^{(d)}\right)^\top, m^{(d)}, \frac{M}{m^{(d)}}\right)$ (equation (3-27b))
4: **end for**
5: $\mathbf{K_{u,u}v_f} = \text{vec}\left(\mathbf{V}^{(0)}\right)$

3-4-2 Predictive Mean Estimation Using Conjugate Gradients

The linear system $\mathbf{A}\boldsymbol{\alpha} = \mathbf{Y}^{(d)}$ of equation (3-24) and the corresponding D-SKI alternative for scalar potential modeling are solved using iterative conjugate gradients. The conjugate gradient method is a Krylov subspace method, which, for the system $\mathbf{A}\boldsymbol{\alpha} = \mathbf{Y}^{(d)}$, is based on the Krylov subspace

$$\mathcal{K}(\mathbf{A}, \mathbf{Y}^{(d)}) = \text{span}\left\{\mathbf{Y}^{(d)}, \mathbf{A}\mathbf{Y}^{(d)}, \ldots, \mathbf{A}^{N-1}\mathbf{Y}^{(d)}\right\}. \tag{3-28}$$

The matrix $\mathbf{A}$ is required to be a symmetric positive-definite matrix, which is given considering $\mathbf{A}$ is a covariance matrix [38]. The optimal solution $\boldsymbol{\alpha}^*$ of the system $\mathbf{A}\boldsymbol{\alpha} = \mathbf{Y}^{(d)}$ is given by the unique minimum of the quadratic function $\phi(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^\top\mathbf{A}\boldsymbol{\alpha} - \boldsymbol{\alpha}^\top\mathbf{Y}^{(d)}$. Iteratively solving linear systems using the conjugate gradient method is faster compared to numerically stable decompositions, such as Cholesky or QR, if $J < N$, where J is the number of iterations. In practice, $J \ll N$ for accurate solutions, although the convergence rate depends on the conditioning of $\mathbf{A}$ [38].

To increase the convergence rate, both sides of $\mathbf{A}\boldsymbol{\alpha} = \mathbf{Y}^{(d)}$ are left-multiplied with a preconditioner $\mathbf{P}^{-1}$, i.e., $\mathbf{P}^{-1}\mathbf{A}\boldsymbol{\alpha} = \mathbf{P}^{-1}\mathbf{Y}^{(d)}$. This preconditioner modifies the system to have superior conditioning. Solving this system for optimal $\boldsymbol{\alpha}^*$ using the preconditioned conjugate gradient method is given by Algorithm 4 with an arbitrary $\boldsymbol{\alpha}_0$; either an approximate initial solution or $\mathbf{0}$ [42]. The loop is constructed such that the algorithm returns $\boldsymbol{\alpha}_J$ when the residual $\mathbf{r}_J^\top\mathbf{z}_J$ is below a desired tolerance. The modified system requires fewer iterations to get the residual below the desired tolerance compared to the unmodified system $\mathbf{A}\boldsymbol{\alpha} = \mathbf{Y}^{(d)}$. However, every iteration is slower due to the computation of $\mathbf{P}^{-1}\mathbf{r}$. By choosing an appropriate preconditioner, $\mathbf{Pz} = \mathbf{r}$ can be solved quickly in each iteration, minimizing the additional time per iteration.

An efficient preconditioner involves the pivoted Cholesky decomposition, which computes a low-rank approximation $\mathbf{L}_L\mathbf{L}_L^\top$ of the positive-definite matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ [43], where $\mathbf{L}_L \in \mathbb{R}^{M \times L}$. Computation of the pivoted Cholesky decomposition has a complexity of $\mathcal{O}(ML^2)$, and, similar to $\mathbf{K}_{\mathbf{u},\mathbf{u}}$, decomposes as a Kronecker product [44], i.e.,

$$\mathbf{L}_L = \bigotimes_{d=1}^{D} \mathbf{L}_{l^{(d)}}^{(d)}, \tag{3-29}$$

where $L = \prod_{d=1}^{D} l^{(d)}$. Computation of the $l^{(d)}$-rank pivoted Cholesky decomposition of $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}$ is described by Algorithm 5 [43, 45], allowing for more efficient computation over D dimensions

Algorithm 4 Preconditioned conjugate gradient method

Input: $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)} \in \mathbb{R}^{m^{(d)} \times m^{(d)}}$ for $d = 1$ to D, $\mathbf{W}_\mathbf{f} \in \mathbb{R}^{N \times M}$, $\mathbf{Y}^{(d)} \in \mathbb{R}^N$, $\boldsymbol{\alpha}_0 \in \mathbb{R}^N$, $\mathbf{L}_L \in \mathbb{R}^{M \times L}$
Output: $\boldsymbol{\alpha}^* \in \mathbb{R}^N$

compute $\mathbf{r}_0 = \mathbf{Y}^{(d)} - \mathbf{A}\boldsymbol{\alpha}_0$ (equations (3-23) and (3-26), and Algorithm 3)
$\mathbf{z}_0 = \mathbf{P}^{-1}\mathbf{r}_0$ (equation (3-31))
$\mathbf{p}_0 = \mathbf{z}_0$
for $i = 1$ to N **do**
    compute $\mathbf{A}\mathbf{p}_i$ (equations (3-23) and (3-26), and Algorithm 3)
    $\rho_i = \frac{\mathbf{r}_i^\top \mathbf{z}_i}{\mathbf{p}_i^\top \mathbf{A}\mathbf{p}_i}$
    $\boldsymbol{\alpha}_{i+1} = \boldsymbol{\alpha}_i + \rho_i \mathbf{p}_i$
    $\mathbf{r}_{i+1} = \mathbf{r}_i - \rho_i \mathbf{A}\mathbf{p}_i$
    $\mathbf{z}_{i+1} = \mathbf{P}^{-1}\mathbf{r}_{i+1}$ (equation (3-31))
    if $\mathbf{r}_{i+1}^\top \mathbf{z}_{i+1}$ is sufficiently small **then**
        exit loop
    end if
    $\mathbf{p}_{i+1} = \mathbf{z}_{i+1} + \frac{\mathbf{r}_{i+1}^\top \mathbf{z}_{i+1}}{\mathbf{r}_i^\top \mathbf{z}_i}\mathbf{p}_i$
end for
$\boldsymbol{\alpha}^* = \boldsymbol{\alpha}_{i+1}$

through equation (3-29). By choosing $\mathbf{P}$ to approximate $\mathbf{A}$, but with low-rank approximation $\mathbf{L}_L\mathbf{L}_L^\top$, the preconditioner is given by

$$\mathbf{P} = \left(\mathbf{W}_\mathbf{f}\mathbf{L}_L\mathbf{L}_L^\top\mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2\mathbf{I}\right). \tag{3-30}$$

Computing $\mathbf{P}^{-1}\mathbf{r}$ still retains a computational complexity similar to $\mathcal{O}(M^3)$ due to the matrix inverse. The Woodbury matrix identity is used to reduce this computational complexity [38, 45], which efficiently computes $\mathbf{P}^{-1}\mathbf{r}$ according to

$$\begin{aligned}\mathbf{P}^{-1}\mathbf{r} &= \left(\mathbf{W}_\mathbf{f}\mathbf{L}_L\mathbf{L}_L^\top\mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2\mathbf{I}\right)^{-1}\mathbf{r} \\ &= \frac{1}{\sigma_\mathrm{n}^2}\mathbf{r} - \frac{1}{\sigma_\mathrm{n}^4}\mathbf{W}_\mathbf{f}\mathbf{L}_L\left(\mathbf{I} + \frac{1}{\sigma_\mathrm{n}^2}\mathbf{L}_L^\top\mathbf{W}_\mathbf{f}^\top\mathbf{W}_\mathbf{f}\mathbf{L}_L\right)^{-1}\mathbf{L}_L^\top\mathbf{W}_\mathbf{f}^\top\mathbf{r}.\end{aligned} \tag{3-31}$$

As the linear system is now of size $L \times L$, the computational complexity of computing $\mathbf{P}^{-1}\mathbf{r}$ is reduced to $\mathcal{O}(ML^2)$, dominated by matrix multiplications. This computational complexity is negligible if L is chosen reasonably low [45], such that iteratively solving $\mathbf{A}^{-1}\mathbf{Y}^{(d)}$ to find $\boldsymbol{\alpha}^*$ is more efficient using the preconditioned version of the conjugate gradient method. The computation of the predictive mean of equation (3-21a) is then straightforward, computed as $\mathbf{W}_*\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_\mathbf{f}^\top\boldsymbol{\alpha}^*$. As $\mathbf{a} = \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_\mathbf{f}^\top\boldsymbol{\alpha}^* \in \mathbb{R}^M$ is only based on the training data, it can be efficiently precomputed using the structure of $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and $\mathbf{W}_\mathbf{f}$ without knowledge of the test inputs. The only requirement for the test inputs is that they are confined within the same inducing input grid as the training inputs. Computing $\mathbf{a}$ has an associated computational complexity of $\mathcal{O}(J(N + M\sum_{d=1}^D m^{(d)}))$ if L is chosen to be appropriately low. The predictive mean of the scalar potential model of equation (3-22a) can be computed similarly, alternatively solving $(\partial\mathbf{A})^{-1}\,\mathrm{vec}(\mathbf{Y}^\top)$ using the preconditioned conjugate gradient method and substituting $\partial\mathbf{W}_\mathbf{f}$ for $\mathbf{W}_\mathbf{f}$.

Algorithm 5 $l^{(d)}$-rank pivoted Cholesky decomposition of $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}$ [45]

Input: $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)} \in \mathbb{R}^{m^{(d)} \times m^{(d)}}$, $l^{(d)}$ subject to $l^{(d)} < m^{(d)}$
Output: $\mathbf{L}_{l^{(d)}} \in \mathbb{R}^{m^{(d)} \times l^{(d)}}$

$\mathbf{S}_0 = \mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}$
$\mathbf{L}_{l^{(d)}} = [\emptyset]$
for $k = 1$ to $l^{(d)}$ **do**
  find S_{jj}, the maximum diagonal element of $\mathbf{S}_{k-1}$
  swap rows and columns of $\mathbf{S}_{k-1}$ with $\boldsymbol{\pi}_k \mathbf{S}_{k-1} \boldsymbol{\pi}_k$ such that S_{jj} is the upper-left entry
  parameterize $\mathbf{S}_{k-1}$ as $\begin{bmatrix} S_{11} & \mathbf{S}_{21}^\top \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{bmatrix}$, where $\mathbf{S}_{21} \in \mathbb{R}^{m^{(d)}-k}$ and $\mathbf{S}_{22} \in \mathbb{R}^{\left(m^{(d)}-k\right) \times \left(m^{(d)}-k\right)}$
  $\mathbf{b}_k = \begin{bmatrix} \sqrt{S_{11}} \\ \frac{1}{\sqrt{S_{11}}} \mathbf{S}_{21} \end{bmatrix}$
  $\mathbf{S}_k = \mathbf{S}_{22} - \frac{1}{S_{11}} \mathbf{S}_{21} \mathbf{S}_{21}^\top$
  $\hat{\mathbf{b}}_k = \begin{bmatrix} \mathbf{0} \\ \mathbf{b}_k \end{bmatrix}$ such that $\hat{\mathbf{b}}_k \in \mathbb{R}^{m^{(d)}}$
  $\hat{\boldsymbol{\pi}}_k = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\pi}_k \end{bmatrix}$ such that $\hat{\boldsymbol{\pi}}_k \in \mathbb{R}^{m^{(d)} \times m^{(d)}}$
  $\mathbf{P}_{\hat{\boldsymbol{\pi}},k} = \prod_{j=1}^{k} \hat{\boldsymbol{\pi}}_j$
  $\mathbf{L}_{l^{(d)}} = \begin{bmatrix} \mathbf{L}_{l^{(d)}} & \mathbf{P}_{\hat{\boldsymbol{\pi}},k} \hat{\mathbf{b}}_k \end{bmatrix}$
end for

3-4-3 Predictive Variance Estimation Using Lanczos Variance Estimates

For the efficient computation of the predictive variances of the shared and scalar potential models within the SKI framework (equations (3-21b) and (3-22b)), fast computation of the precomputable $\mathbf{C}$ and $\partial\mathbf{C}$ matrices is required. While using the preconditioned conjugate gradient method to efficiently compute $\mathbf{A}^{-1}\mathbf{Y}^{(d)}$ and $(\partial\mathbf{A})^{-1}\,\mathrm{vec}(\mathbf{Y}^\top)$ for the predictive means works very well, sequentially solving $\mathbf{A}^{-1}(\mathbf{W_f}\mathbf{K_{u,u}})$ and $(\partial\mathbf{A})^{-1}((\partial\mathbf{W_f})\mathbf{K_{u,u}})$ for each column is not particularly efficient. An efficient method for estimation of the $\mathbf{C}$ and $\partial\mathbf{C}$ matrices is thus desired. In one such method researched within the SKI framework [40], the variance is stochastically estimated by drawing samples from the predictive distributions [46]. While this approach can reduce the computational complexity associated with the computation of the predictive variances, it introduces significant accuracy losses.

The LanczOs Variance Estimates (LOVE) approach aims to address these shortcomings [20]. This approach is based on the Lanczos tridiagonalization algorithm, which is used to find a low-rank approximation for the $\mathbf{A}$ and $\partial\mathbf{A}$ matrices by computing an orthonormal basis of a Krylov subspace [38]. It is thus also based on MVMs, which can be efficiently computed as described in Section 3-4-1. The Krylov subspaces used in LOVE are based on the probe vectors $\mathbf{q}_{\text{probe}} = \mathbf{W_f}\mathbf{K_{u,u}}\mathbf{1} \in \mathbb{R}^N$ and $\partial\mathbf{q}_{\text{probe}} = (\partial\mathbf{W_f})\mathbf{K_{u,u}}\mathbf{1} \in \mathbb{R}^{DN}$, given by $\mathcal{K}(\mathbf{A}, \mathbf{q}_{\text{probe}})$ and $\mathcal{K}(\partial\mathbf{A}, \partial\mathbf{q}_{\text{probe}})$ [20]. Similar to the use of the conjugate gradient method to compute the predictive mean, only estimation of the predictive variance for the shared model of equation (3-21b) is described throughout this section. For the predictive variance for the scalar potential model of equation (3-22b), simply substitute $\partial\mathbf{A}$ for $\mathbf{A}$ and $\partial\mathbf{W_f}$ for $\mathbf{W_f}$.

For the symmetric covariance matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, the Lanczos tridiagonalization algorithm factorizes $\mathbf{A}$ as $\mathbf{QTQ}^\top$ if ran for N iterations [38]. In this factorization, $\mathbf{T} \in \mathbb{R}^{N \times N}$ is a symmetric tridiagonal matrix whose eigenvalues correspond to those of $\mathbf{A}$, whereas $\mathbf{Q} \in \mathbb{R}^{N \times N}$ is an orthogonal matrix. While the Lanczos tridiagonalization algorithm results in a useful factorization, execution is expensive, up to the point where it is prohibitive in implementations where computation speed is of importance.

The advantage of the Lanczos tridiagonalization algorithm comes from the properties of the algorithm. As the algorithm is executed iteratively and the eigenvalues of $\mathbf{T}$ converge to the extreme eigenvalues of $\mathbf{A}$ first over iterations, a sufficient factorization $\mathbf{A} \approx \mathbf{Q}_T \mathbf{T}_T \mathbf{Q}_T^\top$ may be computed given $T \ll N$ [47]. Only the first T orthonormal vectors $\mathbf{Q}_T \in \mathbb{R}^{N \times T}$ of $\mathbf{Q}$ are then computed, together with all α and β coefficients of $\mathbf{T}_T \in \mathbb{R}^{T \times T}$ [38]. The sparse tridiagonal structure of $\mathbf{T}_T$ is given by

$$\mathbf{T}_T = \begin{bmatrix} \alpha_1 & \beta_1 & & \dots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{T-1} \\ 0 & \dots & & \beta_{T-1} & \alpha_T \end{bmatrix}. \tag{3-32}$$

The matrices $\mathbf{Q}_T$ and $\mathbf{T}_T$ are used to efficiently estimate the predictive variance through the estimation of $\mathbf{C}$. Using the factorization $\mathbf{A} \approx \mathbf{Q}_T \mathbf{T}_T \mathbf{Q}_T^\top$, $\mathbf{C}$ is approximated as

$$\begin{aligned} \mathbf{C} &= \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \left(\mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top + \sigma_\mathrm{n}^2 \mathbf{I} \right)^{-1} \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \\ &\approx \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \left(\mathbf{Q}_T \mathbf{T}_T^{-1} \mathbf{Q}_T^\top \right) \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}} \\ &= \underbrace{\mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \mathbf{Q}_T}_{\mathbf{R}^\top} \underbrace{\mathbf{T}_T^{-1} \mathbf{Q}_T^\top \mathbf{W}_\mathbf{f} \mathbf{K}_{\mathbf{u},\mathbf{u}}}_{\mathbf{R}'}. \end{aligned} \tag{3-33}$$

The Lanczos tridiagonalization algorithm used to find $\mathbf{Q}_T$ and $\mathbf{T}_T$ for T iterations is given by Algorithm 6 [20, 38]. As mentioned previously, iterations of the Lanczos algorithm are based on MVMs, thus enabling efficient computation through the exploitation of the favorable Kronecker structure present in $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ as described in Section 3-4-1. While the Lanczos algorithm computes the orthonormal vectors of $\mathbf{Q}_T$, roundoff errors occur during the orthogonalization of $\mathbf{q}_t$ in numerical implementations using inexact arithmetic (e.g., floating-point arithmetic). These roundoff errors result in numerical instability of the Lanczos algorithm, which can be resolved using reorthogonalization. Through full reorthogonalization of $\mathbf{q}_t$ against all receding vectors, $\mathbf{Q}_{t-1}$, orthogonality and thus numerical stability are ensured [48].

For efficient computation of $\mathbf{R}^\top = \mathbf{K}_{\mathbf{u},\mathbf{u}} \mathbf{W}_\mathbf{f}^\top \mathbf{Q}_T \in \mathbb{R}^{M \times T}$, fast MVMs can be exploited sequentially for all T columns of $\mathbf{Q}_T$. Subsequently solving $\mathbf{R}' = \mathbf{T}_T^{-1} \mathbf{R}^\top$ can be performed efficiently given $\mathbf{T}_T$ is a very sparse symmetric tridiagonal matrix. Estimation of $\mathbf{C}$ through $\mathbf{R}$ and $\mathbf{R}'$ has an associated computational complexity of $\mathcal{O}(2T(N + M \sum_{d=1}^{D} m^{(d)}))$ for magnetic field modeling [20]. In numerical implementations, the complexity is higher due to the full reorthogonalization required for the Lanczos tridiagonalization algorithm, also scaling linearly with the number of training points N and Lanczos iterations T. Similar to the computation of the predictive mean, the matrices $\mathbf{R}$ and $\mathbf{R}'$ can be precomputed as they are only based

on the training data. For fast estimation of the predictive variances of the shared and scalar potential models for magnetic field modeling, $\mathbf{R}$ and $\mathbf{R}'$ can be used to rewrite the predictive variance equations for fast test-time predictions, as described in the next section.

Algorithm 6 Lanczos tridiagonalization for T iterations (with full reorthogonalization)

Input: $\mathbf{K_{u,u}} \in \mathbb{R}^{m^{(d)} \times m^{(d)}}$ for $d = 1$ to D, $\mathbf{W_f} \in \mathbb{R}^{N \times M}$, σ_n, T
Output: $\mathbf{T}_T \in \mathbb{R}^{T \times T}$, $\mathbf{Q}_T \in \mathbb{R}^{N \times T}$

compute $\mathbf{q}_\text{probe} = \mathbf{W_f K_{u,u} 1}$ using Kronecker MVM (equation (3-26) and Algorithm 3)
$\mathbf{r}_0 = \mathbf{q}_\text{probe}$
$\beta_0 = ||\mathbf{q}_\text{probe}||_2$
$\mathbf{Q}_0 = [\emptyset]$
for $t = 1$ to T **do**
    $\mathbf{q}_t = \mathbf{r}_{t-1}/\beta_{t-1}$
    if $t > 1$ **then** `// full reorthogonalization`
        $\mathbf{q}_t = \mathbf{q}_t - \mathbf{Q}_{t-1}\mathbf{Q}_{t-1}^\top \mathbf{q}_t$
        $\mathbf{q}_t = \mathbf{q}_t / ||\mathbf{q}_t||_2$
    end if
    $\mathbf{Q}_t = [\mathbf{Q}_{t-1} \quad \mathbf{q}_t]$
    compute $\mathbf{A}\mathbf{q}_t$ using Kronecker MVM (equation (3-26) and Algorithm 3)
    $\alpha_t = \mathbf{q}_t^\top \mathbf{A}\mathbf{q}_t$
    $\mathbf{r}_t = \mathbf{A}\mathbf{q}_t - \alpha_t \mathbf{q}_t - \beta_{t-1}\mathbf{q}_{t-1}$
    $\beta_t = ||\mathbf{r}_t||_2$
end for
construct $\mathbf{T}_T$ using all α_t and β_t (excluding β_0)

3-4-4 Fast Test-Time Predictions of the Magnetic Field

In the previous sections, the SKI framework is used for efficient computation of the precomputable matrices for magnetic field modeling based on training data only. As mentioned in Section 3-4-2, the test inputs are only required to be confined within the same inducing input grid as the training inputs. With the preconditioned conjugate gradient method and LOVE, the structure in $\mathbf{K_{u,u}}$ and the sparse interpolation matrices is exploited for efficient computation of $\mathbf{a} = \mathbf{K_{u,u}} \mathbf{W_f}^\top \mathbf{A}^{-1} \mathbf{Y}^{(d)}$, $\mathbf{R}$, and $\mathbf{R}'$. For the fast computation of the predictive mean, $\mathbf{a}$ only needs to be left-multiplied with $\mathbf{W}_*$ for the shared model. Fast computation of the predictive variance of the shared model can be achieved by rewriting equation (3-21b) using $\mathbf{R}$ and $\mathbf{R}'$, expressed as

$$\mathbb{V}\left[\mathbf{h}^{(d)}(\mathbf{X}_*)\right] = \mathbf{W}_* \mathbf{K_{u,u}} \mathbf{W}_*^\top - \mathbf{W}_* \underbrace{\mathbf{K_{u,u}} \mathbf{W_f}^\top \left(\mathbf{W_f K_{u,u}} \mathbf{W_f}^\top + \sigma_\text{n}^2 \mathbf{I}\right)^{-1} \mathbf{W_f K_{u,u}}}_{\mathbf{C}} \mathbf{W}_*^\top \tag{3-34}$$

$$\approx \mathbf{W}_* \mathbf{K_{u,u}} \mathbf{W}_*^\top - \mathbf{W}_* \mathbf{R}^\top \mathbf{R}' \mathbf{W}_*^\top \tag{3-35}$$

$$= \mathbf{W}_* \mathbf{K_{u,u}} \mathbf{W}_*^\top - \left(\mathbf{R}\mathbf{W}_*^\top\right)^\top \mathbf{R}' \mathbf{W}_*^\top \tag{3-36}$$

A significant computational bottleneck for magnetic field modeling using large numbers of inducing points may be the computation of $\mathbf{W}_* \mathbf{K_{u,u}} \mathbf{W}_*^\top$, which, for large numbers of inducing

and test points, may be computationally intractable. The full computation can be avoided however, as the predictive variances are given by just the diagonal of the test covariance matrix. To this end, the diagonal of the test covariance matrix can be computed through the expressions

$$\operatorname{diag}\left(\mathbf{W}_*\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{W}_*^\top\right)_i = \mathbf{w}_{*,i}(\mathbf{x}_{*,i})\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{w}_{*,i}(\mathbf{x}_{*,i})^\top, \tag{3-37a}$$

and

$$\operatorname{diag}\left((\partial\mathbf{W}_*)\mathbf{K}_{\mathbf{u},\mathbf{u}}(\partial\mathbf{W}_*)^\top\right)_i = \operatorname{diag}\left(\nabla\mathbf{w}_{*,i}(\mathbf{x}_{*,i})\mathbf{K}_{\mathbf{u},\mathbf{u}}\nabla\mathbf{w}_{*,i}(\mathbf{x}_{*,i})^\top\right). \tag{3-37b}$$

Explicit computation of the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}} \in \mathbb{R}^{M\times M}$ can be avoided as the rows of $\mathbf{W}_*$ and $\partial\mathbf{W}_*$ are computed as a Kronecker product over the interpolation weights per dimension (equations (3-10) and (3-12)). Consequently, $\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{w}_*^\top$ can be computed as a Kronecker product over grid dimensions. Its computation is given by

$$\begin{aligned}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{w}_*^\top &= \left(\bigotimes_{d=1}^{D}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}\right)\left(\bigotimes_{d=1}^{D}\left(\mathbf{w}_*^{(d)}\right)^\top\right)\\ &= \bigotimes_{d=1}^{D}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{(d)}\left(\mathbf{w}_*^{(d)}\right)^\top,\end{aligned} \tag{3-38}$$

where $\mathbf{w}_*^{(d)}$ only contains the interpolation weights computed for dimension d. The test covariance matrix computed with $\partial\mathbf{W}_*$ can be computed similarly, exploiting the Kronecker formulations in $\nabla\mathbf{w}_*$ as described by equation 3-12.

As mentioned in Sections 3-4-2 and 3-4-3, efficient computation of the predictive distribution for the scalar potential model is achieved by substituting $\partial\mathbf{A}$ for $\mathbf{A}$ and $\partial\mathbf{W}_\mathbf{f}$ for $\mathbf{W}_\mathbf{f}$ during training computations (conjugate gradients and LOVE). For fast test-time predictions, $\partial\mathbf{W}_*$ can be substituted for $\mathbf{W}_*$ for the multiplications with $\partial\mathbf{R}$ and $\partial\mathbf{R}'$. The method presented in this chapter can be used to efficiently model curl-free vector fields using the shared and scalar potential models. In Chapter 4, the performance of the method is studied using simulations and experiments with real magnetic field measurements. The collection of the magnetic field measurements for the experiments is described in the next section.

3-5 Collecting Magnetic Field Measurements

To show the applicability of the described approach for the estimation of magnetic fields with the shared and scalar potential models, experiments are conducted with magnetic field measurements collected using the MVN Link Suit. The experiments are presented in Sections 4-3 and 4-4. As stated in Section 2-4, the MVN Link Suit and its accompanying software provide the magnetic field measurements with the corresponding position and orientation estimates of 17 sensors at 240 Hz. The position and orientation estimates are dependent on the processing mode used: *live* or *reprocessed HD*. As the *reprocessed HD* mode provides more consistent estimates and sequential estimation of the magnetic field is not desired, this mode is used for the collection of experimental data. Previous work has shown that the use of magnetic field measurements of multiple sensors does not necessarily lead to better estimation of the magnetic field [21] (see Chapter 1), so the magnetic field measurements of one only sensor are

used for experimental results. The sensor located at the pelvis is chosen for the experiments, as it is roughly located at the center of all sensors.

To model magnetic fields using Gaussian process regression based on suit data, the magnetic field measurements and their corresponding position estimates are required in a consistent frame of reference, the so-called global frame $\mathcal{G}$. Conveniently, the position estimates provided by the suit are already given in the global frame, whereas the magnetic field measurements need to be mapped from the pelvis sensor frame $\mathcal{S}$ to the global frame $\mathcal{G}$ via their orientation estimates. The variables required from Table 2-2, recorded at measurement index $i \in \{1, 2, \ldots, N\}$, are denoted by

- ${}_{\mathcal{G}}\mathbf{x}_i \in \mathbb{R}^3$ denotes the position estimate of the pelvis sensor,
- ${}_{\mathcal{S}}\mathbf{y}_i \in \mathbb{R}^3$ denotes the three-dimensional magnetic field measurement of the pelvis sensor in arbitrary units, interpreted as a scaled measurement of the **H**-field,
- ${}_{\mathcal{G}}\mathbf{q}_i^{\mathcal{S}} \in \mathbb{R}^4$ denotes the orientation estimate of the pelvis sensor as a unit quaternion, describing the rotation from pelvis sensor frame $\mathcal{S}$ to global frame $\mathcal{G}$.

To map a magnetic field measurement ${}_{\mathcal{S}}\mathbf{y}_i$ from the pelvis sensor frame $\mathcal{S}$ to the global frame $\mathcal{G}$, a 3×3 rotation matrix is used, ${}_{\mathcal{G}}\mathbf{R}_i^{\mathcal{S}}$. The measurement in the global frame is given by

$$ {}_{\mathcal{G}}\mathbf{y}_i = {}_{\mathcal{G}}\mathbf{R}_i^{\mathcal{S}} {}_{\mathcal{S}}\mathbf{y}_i, \tag{3-39} $$

where the 9-entry rotation matrix $\mathbf{R}$ can be constructed from a 4-entry quaternion $\mathbf{q} \in \mathbb{R}^4$ [49], defining $\mathbf{q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^\top$. The quaternion-derived rotation matrix is given by

$$ \mathbf{R}(\mathbf{q}) = \begin{bmatrix} 2q_0^2 + 2q_1^2 - 1 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & 2q_0^2 + 2q_2^2 - 1 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & 2q_0^2 + 2q_3^2 - 1 \end{bmatrix}. \tag{3-40} $$

The position estimates and magnetic field measurements of the pelvis sensor in the global frame are then collected into the input matrix $\mathbf{X_f} \in \mathbb{R}^{N \times 3}$ and measurement matrix $\mathbf{Y} \in \mathbb{R}^{N \times 3}$, respectively given by

$$ \mathbf{X_f} = \begin{bmatrix} \mathbf{X_f}^{(1)} & \mathbf{X_f}^{(2)} & \mathbf{X_f}^{(3)} \end{bmatrix} = \begin{bmatrix} {}_{\mathcal{G}}\mathbf{x}_1^\top \\ {}_{\mathcal{G}}\mathbf{x}_2^\top \\ \vdots \\ {}_{\mathcal{G}}\mathbf{x}_N^\top \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} \mathbf{Y}^{(1)} & \mathbf{Y}^{(2)} & \mathbf{Y}^{(3)} \end{bmatrix} = \begin{bmatrix} {}_{\mathcal{G}}\mathbf{y}_1^\top \\ {}_{\mathcal{G}}\mathbf{y}_2^\top \\ \vdots \\ {}_{\mathcal{G}}\mathbf{y}_N^\top \end{bmatrix}. \tag{3-41} $$

These matrices are used as the input matrix and measurement matrix for the estimation of the magnetic field using the shared and scalar potential models in the experiments with real magnetic field data as described throughout this chapter. An important property of data sets used in the experiments is that they do not contain loop closures, meaning the pelvis sensor does not return to previously visited locations. This is important as drift present in the position estimates can introduce inconsistencies in the data, possibly having undesirable effects on the estimation of the magnetic field (see Chapter 1). Experiments with magnetic field data collected with the suit are presented in Sections 4-3 and 4-4.

Chapter 4

Experimental Results

In this chapter, experiments are presented to show the applicability of the SKI framework to the shared and scalar potential models stated in Section 2-3 for scalable magnetic field modeling. For the results, the shared and scalar potential models are estimated using the SKI framework as described in Chapter 3. The accuracy and efficiency of magnetic field modeling are studied through simulations and magnetic field measurements collected with the MVN Link Suit, answering the second research subquestion. For the third research subquestion, the scalability of the approach is investigated through two large-scale experiments with data sets that are too large for full Gaussian process regression. As stated in Section 2-3, the shared model considers the magnetic field to be a three-dimensional vector field with independent components and shared hyperparameters, given by equation (2-13). The scalar potential model considers the magnetic field to be the negative gradient of the magnetic scalar potential function φ, given by equation (2-15). The magnetic field components then satisfy the curl-free constraint imposed by Maxwell's equations. All implementation is done in the Julia programming language on a 2016 HP ZBook Studio G3 laptop (Intel Core i7 @ 2.60 GHz, 8 GB RAM).

For interpretation of the results, the metrics used to quantify the quality of trained models are introduced first in Section 4-1. The conducted simulations are presented in Section 4-2, which show the SKI framework accurately and efficiently approximates the shared and scalar potential models under similar simulated circumstances. A first experiment with magnetic field measurements collected with the suit is presented in Section 4-3, used to show the SKI framework can be used to accurately and efficiently model magnetic fields. Lastly, to demonstrate the scalability of the approach for magnetic field modeling, the two large-scale experiments are presented in Section 4-4.

4-1 Experiment Metrics

Throughout this chapter, four metrics are used to assess the accuracy and efficiency of trained magnetic field models. These metrics are the root-mean-square error (RMSE), the mean standardized log loss (MSLL), the relative error (RE), and the computation time. The first three

metrics are used to determine the accuracy of the approximated magnetic field models for the second research subquestion. The computation time is used to determine the efficiency of magnetic field modeling using the SKI framework for the last of the three research subquestions.

The RMSE is used to compare the shared and scalar potential models trained on the same data set and is given by

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N_*} \left(\mathbb{E}\left[f(\mathbf{x}_{*,i})\right] - f_{*,i}\right)^2}{N_*}}, \tag{4-1}$$

where $\mathbb{E}\left[f(\mathbf{x}_{*,i})\right]$ is the predictive mean and $f_{*,i}$ is the reference value at $\mathbf{x}_{*,i}$.

As the RMSE does not consider the predictive variances to quantify the quality of the model, the MSLL is introduced. This metric computes the average negative log probability of the predicted test point considering the model [10], thus including the predictive variance. It is given by

$$\text{MSLL} = \frac{1}{2N_*} \sum_{i=1}^{N_*} \left(\log\left(2\pi\sigma_{*,i}^2\right) + \frac{\left(\mathbb{E}\left[f(\mathbf{x}_{*,i})\right] - f_{*,i}\right)^2}{\sigma_{*,i}^2} \right), \tag{4-2}$$

where $\sigma_{*,i} = \mathbb{V}\left[f(\mathbf{x}_{*,i})\right] + \sigma_{\text{n}}^2$.

The relative error is used to quantify the approximation error between magnetic field estimation using full Gaussian process regression and approximation within the SKI framework. The relative error is formulated as

$$\text{RE} = \frac{||\mathbb{E}\left[\mathbf{f}(\mathbf{X}_*)\right] - \mathbf{f}_*||}{||\mathbf{f}_*||}, \tag{4-3}$$

where $\mathbb{E}\left[\mathbf{f}(\mathbf{X}_*)\right]$ contains the predictive means of all test points and $\mathbf{f}_*$ contains the reference values at all test points.

Lastly, the time required to estimate the magnetic field using the shared and scalar potential models is computed. This can be compared to the computation time of full Gaussian process regression, to quantify the efficiency of magnetic field modeling using the SKI framework. The computation time is considered to consist of the computation of the relevant covariance matrices and the sparse interpolation matrices, precomputation of the matrices based on training data using the preconditioned conjugate gradient method and LOVE, and predictions based on the test inputs. All computation times are determined using Julia's BenchmarkTools package.

4-2 Simulated Estimation of a Curl-Free Vector Field

To show the SKI framework can be used to accurately and efficiently approximate the shared and scalar potential models for magnetic field modeling, simulations are conducted under similar circumstances. To this end, a three-dimensional curl-free vector field, which the magnetic field is assumed to be, is simulated in this experiment using synthetic data generated from an arbitrary curl-free vector field. The scalar potential function of this arbitrary curl-free vector field ($\mathbf{x} \in \mathbb{R}^3$) is given by

$$\varphi_{\text{sim.}}(\mathbf{x}) = x^{(1)} \left(x^{(2)}\right)^2 \left(x^{(3)}\right)^3, \tag{4-4}$$

with corresponding curl-free vector field $-\nabla\varphi_{\text{sim.}}(\mathbf{x}) \in \mathbb{R}^3$. The curl-free property is easily verified through the expression $\nabla \times (-\nabla\varphi_{\text{sim.}}(\mathbf{x})) = \mathbf{0}$. The domain of interest for simulations is chosen to be the cube of size $[-2.0, 2.0] \times [-2.0, 2.0] \times [-2.0, 2.0]$ with its center in the origin. In the simulations, the curl-free vector field is estimated with both the shared and scalar potential models, based on sampled measurements of $-\nabla\varphi_{\text{sim.}}(\mathbf{x})$ with added white Gaussian noise. The hyperparameters used in the simulations are given by Table 4-1, which are optimized on 1,000 random samples of $-\nabla\varphi_{\text{sim.}}(\mathbf{x}) \in \mathbb{R}^3$ using the full log marginal likelihood functions of equations (2-14) and (2-19).

Table 4-1: Optimized hyperparameters for estimation of the simulated curl-free vector field.

Model	ℓ	σ_{f}	σ_{n}
Shared	2.0132	58.4945	4.9902
Scalar potential	2.7834	345.0215	4.9865

The first simulation, presented in Section 4-2-1, is conducted to show the approximation quality of the SKI framework along with the required computation times as the number of inducing points increases. The second simulation is presented in Section 4-2-2, in which the effects of an increasing data set size on the accuracy and efficiency of estimating the curl-free vector field using the SKI framework are shown. The results of the simulations imply the described approach is useful for scalable magnetic field modeling, which is further demonstrated using real magnetic field data in Sections 4-3 and 4-4.

4-2-1 Effects of an Increasing Number of Inducing Points

In the first simulation, 1,000 random samples are taken of the curl-free vector field $-\nabla\varphi_{\text{sim.}}(\mathbf{x})$ with added white Gaussian noise with standard deviation $\sigma_{\text{n}} = 5.0$. The curl-free vector field is then estimated using both full Gaussian process regression and the SKI framework on an equispaced test grid of size $10 \times 10 \times 10$. The experiment is conducted multiple times with an increasing number of inducing points to show the change in accuracy and efficiency as more inducing points are used. The inducing point grid is of size $m \times m \times m$ for a total of $M = m^3$ inducing points, where $m \in \{8, 9, \ldots, 24, 25\}$. For computation of the predictive mean within the SKI framework using the preconditioned conjugate gradient method (Algorithm 4), a residual tolerance of $\mathbf{r}_{i+1}^{\top}\mathbf{z}_{i+1} = 10^{-8}$ and a pivoted Cholesky decomposition rank of $l^{(d)} = 2$ for $d = 1, 2, 3$ are specified. For estimation of the predictive variance using LOVE, $T = 200$ Lanczos iterations are used (Algorithm 6).

For quantification of the accuracy and the efficiency, the relative errors between full Gaussian process regression and the SKI framework approximation, and the times required for estimation of the curl-free vector field are computed. For consistent quantification of the accuracy and efficiency, the curl-free vector field is estimated 30 times for each number of inducing points using different randomly sampled data sets. The average of the relative errors over the 30 runs with corresponding standard deviations are shown in Figure 4-1 for the shared model and in Figure 4-2 for the scalar potential model. The use of more inducing points results in lower relative errors and thus better estimates. Relative errors of less than 1% (10^{-2}) are reached quickly, so the approximated estimations of the curl-free vector field are

of good quality. The times required to estimate the vector field are shown in Figure 4-3. For the shared model, the use of the SKI framework with cubic convolution interpolation shows a significant speedup up to a total of 6,000 inducing points used, while using SKI with quintic convolution interpolation does not result in more efficient estimation. The benefits of using the SKI framework are more apparent with the scalar potential model, where even relatively large numbers of inducing points result in lower computation times compared to full Gaussian process regression. As the number of training points in this simulation is relatively low and the SKI framework is expected to result in more significant speedups with more training points, the effects of an increasing data set size are investigated in the next simulation.

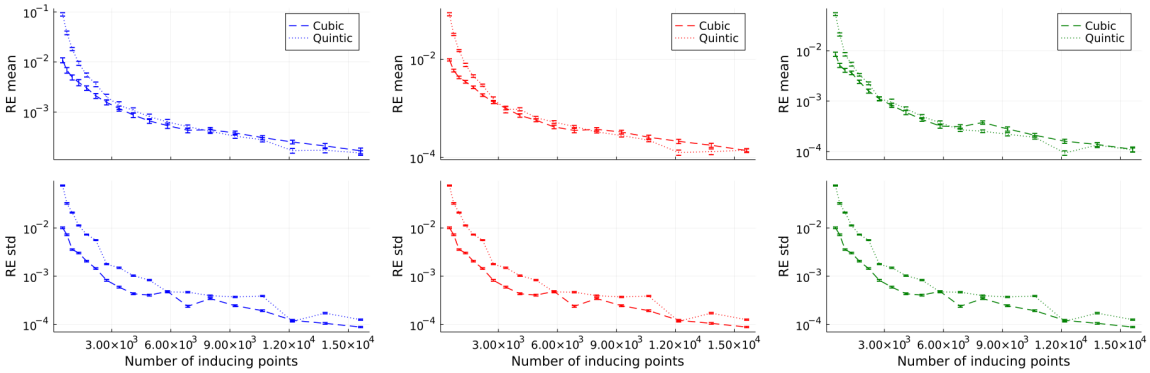
4-2-2 Effects of an Increasing Data Set Size

In the previous simulation, it was shown the shared and scalar potential models are efficiently approximated using the SKI framework with proper accuracy given sufficient inducing points. To investigate the scalability of the approach, the same curl-free vector field is modeled with an increasing number of training points, ranging from $N = 250$ to $N = 10{,}000$ measurements. The training data are generated by taking N samples of the curl-free vector field $-\varphi_{\text{sim.}}(\mathbf{x}) \in \mathbb{R}^3$ with added white Gaussian noise with standard deviation $\sigma_{\text{n}} = 5.0$. The inducing point grid is now fixed at $20 \times 20 \times 20$, for a total of 8,000 inducing points. For the preconditioned conjugate gradient algorithm (Algorithm 4), a residual tolerance of $\mathbf{r}_{i+1}^{\top}\mathbf{z}_{i+1} = 10^{-8}$ is used with a pivoted Cholesky decomposition rank of $l^{(d)} = 5$, along with $T = 200$ Lanczos iterations for the estimation of the predictive variance (Algorithm 6).

To show the effects of the increasing data set size on the accuracy and efficiency of the estimated curl-free vector field, the RMSE and MSLL are determined together with the required computation times. As was done in the previous simulation, the simulation is repeated 30 times with different randomly sampled training data sets for consistent quantification. The average of the RMSEs and MSLLs over the 30 runs with corresponding standard deviations are shown in Figure 4-4 for the shared model and in Figure 4-5 for the scalar potential model. Both metrics decrease as the number of training points increases, indicating better predictions. The scalar potential model results in lower error metrics than the shared model, which is to be expected as the simulated vector field is curl-free. The computation times required to estimate the vector field are shown in Figure 4-6. The use of the SKI framework for faster estimation is beneficial, as the computation times are only similar to full Gaussian process regression for low numbers of training points. Again, using quintic convolution interpolation requires significantly longer computation times compared to cubic convolution interpolation while producing similar error metrics.

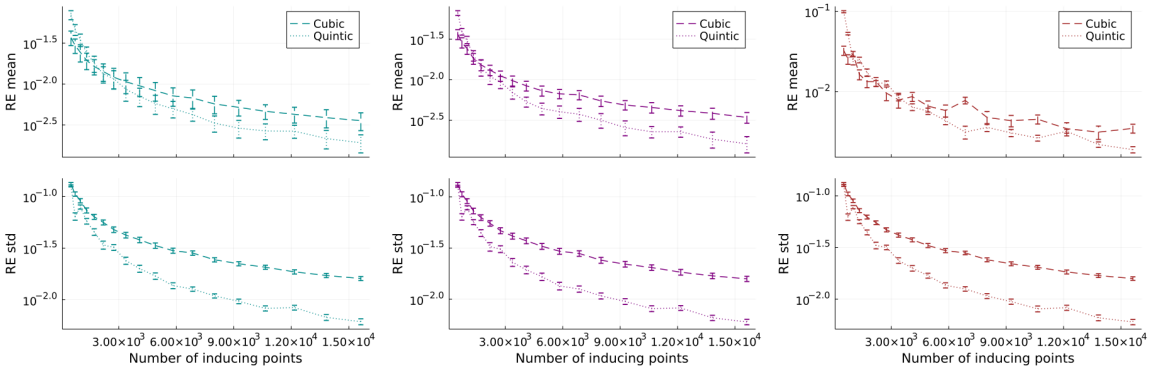
4-3 Accurate and Efficient Magnetic Field Modeling

In the previous section, it was shown the SKI framework can be used to accurately and efficiently approximate the shared and scalar potential models for a simulated curl-free vector field, which the magnetic field is assumed to be. To show the applicability of the SKI framework for efficient magnetic field modeling using real magnetic field data, a moderately sized experiment is conducted with data collected using the MVN Link Suit. The data set consists of $N = 2{,}500$ measurements, allowing for magnetic field estimation using both full Gaussian



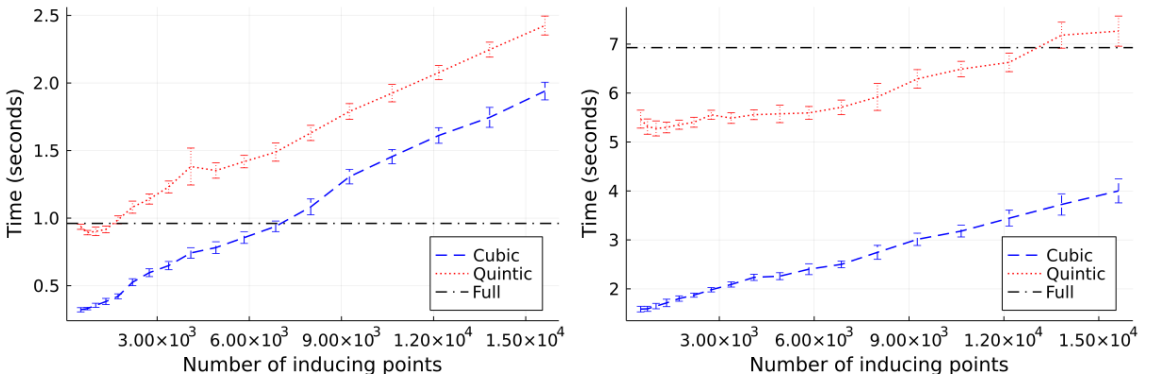
(a) x-component **(b)** y-component **(c)** z-component

Figure 4-1: The relative errors of the SKI approximation for the **shared model** as a function of the number of **inducing points**.



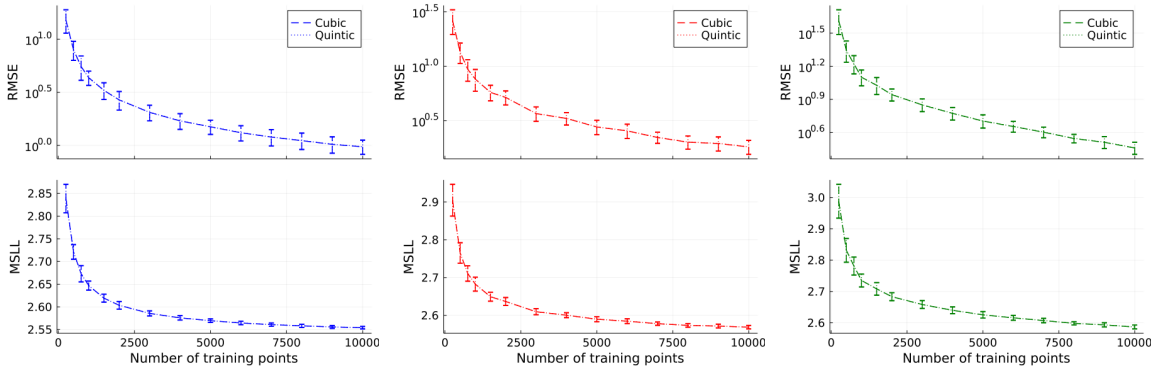
(a) x-component **(b)** y-component **(c)** z-component

Figure 4-2: The relative errors of the SKI approximation for the **scalar potential model** as a function of the number of **inducing points**.



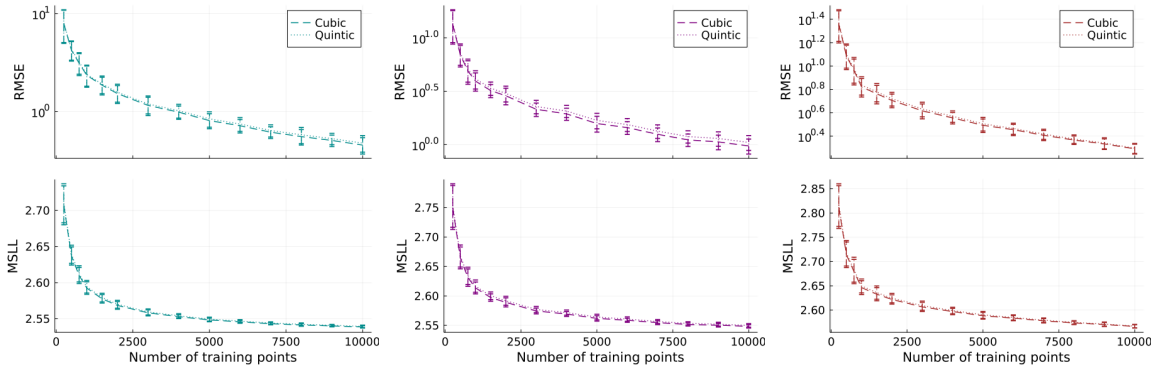
(a) Shared model **(b)** Scalar potential model

Figure 4-3: The average computation times of the first simulation over 30 runs for both models with an increasing number of **inducing points**, including the corresponding standard deviations. Note the discrepancy in the values on the time axis.



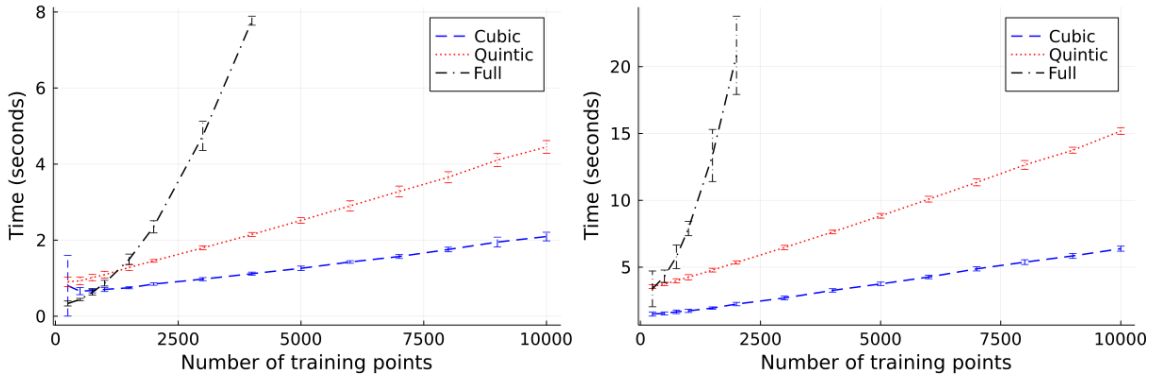
(a) x-component **(b)** y-component **(c)** z-component

Figure 4-4: The RMSE and MSLL of the SKI approximation for the **shared model** as a function of the number of **training points**.



(a) x-component **(b)** y-component **(c)** z-component

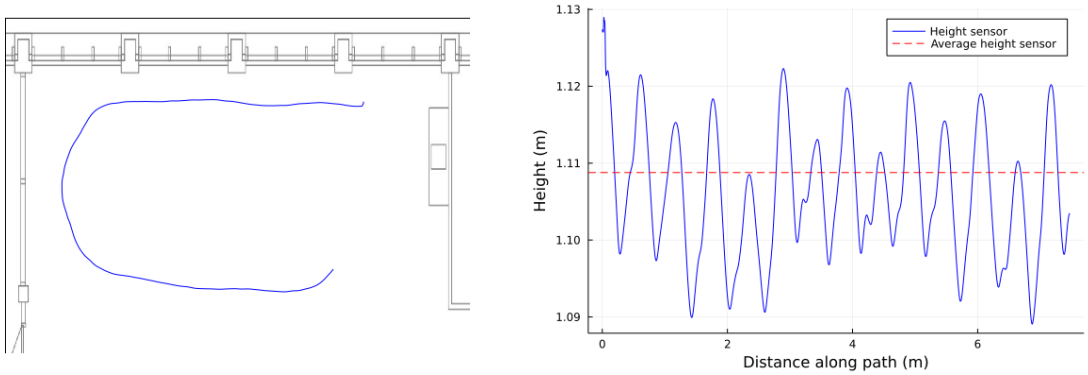
Figure 4-5: The RMSE and MSLL of the SKI approximation for the **scalar potential model** as a function of the number of **training points**.



(a) Shared model **(b)** Scalar potential model

Figure 4-6: The average computation times of the second simulation over 30 runs for both models with an increasing number of **inducing points**, including the corresponding standard deviations. Note the discrepancy in the values on the time axis.

process regression and the SKI framework. For this experiment, the relative errors and the computation times are determined to quantify the accuracy and efficiency of using the SKI framework for magnetic field modeling. The data are collected as described in Section 3-5, measured along a path walked in the motion capture lab of Delft Center for Systems and Control (DCSC, 3mE building), Delft University of Technology. A two-dimensional top view of the walked path and the height of the pelvis sensor with respect to the ground are shown in Figure 4-7. The data set does not contain loop closures to avoid any inconsistencies in the data, which can have undesirable effects on the estimation of the magnetic field as stated in Section 3-5.



(a) Path pelvic sensor **(b)** Height pelvic sensor along path

Figure 4-7: Position estimates of the magnetic field measurements of the pelvis sensor in the motion capture lab of DCSC, Delft University of Technology. The left figure shows the walked path from above. On the right, the height of the pelvis sensor is shown as a function of the distance along the path.

The magnetic field of the motion capture lab is estimated with all 2,500 measurements, using both full Gaussian process regression and the SKI framework. The magnetic field is predicted on a two-dimensional test grid of 50×50 points structured in the square $[-5\,\text{m}, -1\,\text{m}] \times [-1.6\,\text{m}, 1.0\,\text{m}]$ located at a height of 1.1088 m, which is the average of the measured height of the pelvis sensor (see Figure 4-7b). The hyperparameters are determined by maximizing the log marginal likelihood functions for full Gaussian process regression of equations (2-14) and (2-19) with all 2,500 measurements. An overview of the optimized hyperparameters is given in Table 4-2. For estimation of the magnetic field using the SKI framework, the sparse interpolation matrices are computed using cubic convolution interpolation on an inducing point grid of size $40 \times 25 \times 4$, for a total of $M = 4{,}000$ inducing points. Quintic convolution interpolation is avoided as the simulations showed its use only results in slight accuracy gains with significant efficiency losses (see Section 4-2). For the preconditioned conjugate gradient method (Algorithm 4), a residual tolerance of $\mathbf{r}_{i+1}^\top \mathbf{z}_{i+1} = 10^{-8}$ is specified together with a pivoted Cholesky decomposition rank of $l^{(d)} = 6$ for $d = 1, 2, 3$. The predictive variances are estimated using $T = 200$ Lanczos iterations (Algorithm 6).

The magnetic field estimated with the shared model is shown in Figure 4-8. It is easily observed the approximation resembles the magnetic field estimated using full Gaussian process regression well, with few noticeable differences. This is very similar for magnetic field estimation with the scalar potential model, shown in Figure 4-9, although more minor differences can be observed. Note the lack of boundary conditions near the domain regions, not requiring

Table 4-2: Optimized hyperparameters for magnetic field estimation with motion capture suit data.

Model	ℓ	σ_f	σ_n
Shared	0.5319 m	0.0434	0.0266
Scalar potential	0.5799 m	0.0313	0.0266

the predictive mean and variance to revert to the prior mean and zero respectively, as is the case with the established Laplace operator eigenbasis approximation for scalable magnetic field modeling. The approximation errors for both models are quantified in Table 4-3 using the relative error metric. To reduce the approximation errors further, more inducing points can be used. The times required to make the predictions are shown in Table 4-4, which are averaged over 30 runs. The use of the SKI framework results in significant speedups while retaining acceptable accuracy levels. Its use is thus beneficial for magnetic field modeling even for cases where full Gaussian process regression is computationally tractable.

Table 4-3: Approximation errors of the estimated magnetic field of the motion capture lab.

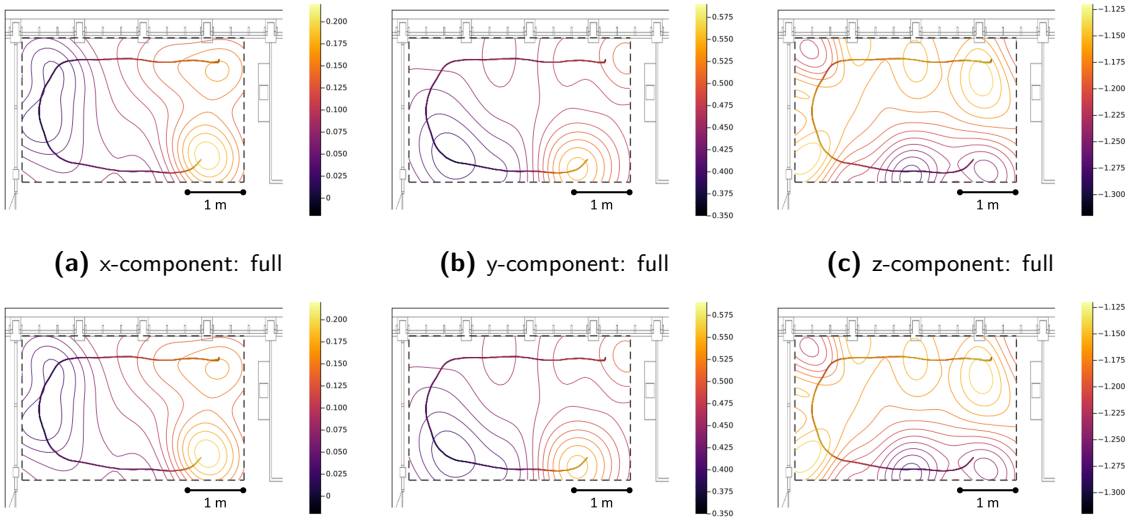
Model	Component	RE mean	RE variance
Shared	x	2.32×10^{-4}	7.09×10^{-4}
	y	1.31×10^{-4}	7.09×10^{-4}
	z	1.22×10^{-4}	7.09×10^{-4}
Scalar potential	x	8.30×10^{-3}	1.41×10^{-2}
	y	4.34×10^{-3}	1.89×10^{-2}
	z	4.31×10^{-4}	6.74×10^{-4}

Table 4-4: Average computation times with corresponding standard deviations of the predictions of the magnetic field in the motion capture lab over 30 runs.

Model	Full	SKI	Average speedup
Shared	7.40 s ± 0.23 s	1.27 s ± 0.03 s	5.83×
Scalar potential	88.53 s ± 2.00 s	7.47 s ± 0.19 s	11.9×

4-4 Large-Scale Magnetic Field Modeling

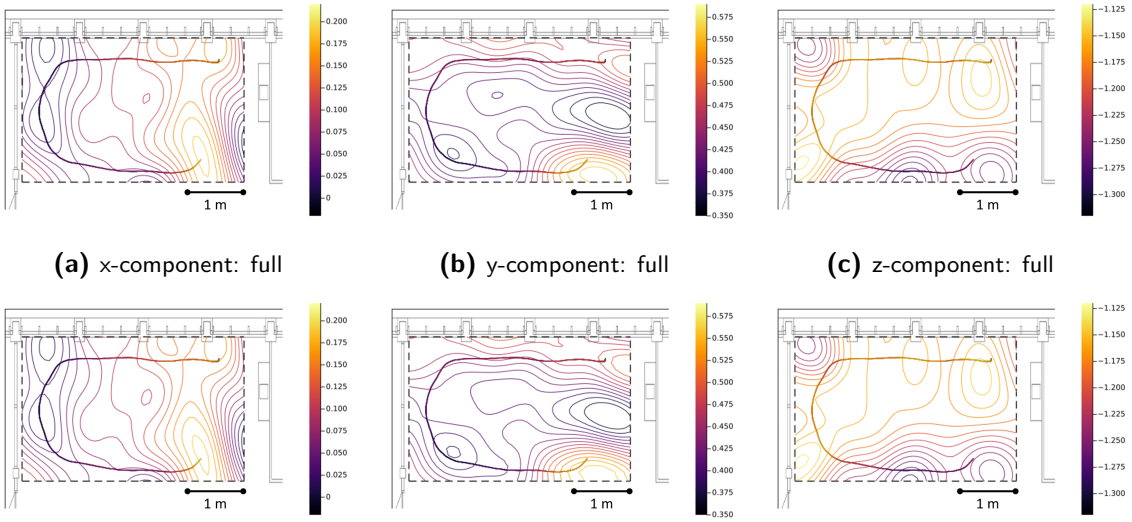
In the previous experiment, it was shown the SKI framework can be used to efficiently estimate magnetic fields using Gaussian process regression while retaining accuracy. The efficiency of regression with the SKI framework allows for the use of larger data sets compared to full Gaussian process regression, which becomes computationally intractable from roughly 10,000 measurements for the shared model and 3,300 for the scalar potential model. This feature of the SKI framework is important for scalable magnetic field modeling, which is emphasized in



(a) x-component: full **(b)** y-component: full **(c)** z-component: full

(d) x-component: SKI **(e)** y-component: SKI **(f)** z-component: SKI

Figure 4-8: The magnetic field of the motion capture lab estimated with the **shared model**. The black dashed box shows the prediction boundaries.



(a) x-component: full **(b)** y-component: full **(c)** z-component: full

(d) x-component: SKI **(e)** y-component: SKI **(f)** z-component: SKI

Figure 4-9: The magnetic field of the motion capture lab estimated with the **scalar potential model**. The black dashed box shows the prediction boundaries.

this section through two experiments. In both experiments, the magnetic field in the hallway of the 3mE building of Delft University of Technology is estimated with two large data sets that are too large for full Gaussian process regression. The experiments are conducted with the full data sets (no splitting for faster regression), so no transition regions are present in the estimated magnetic fields. The data sets are collected as described in Section 3-5. The first experiment is presented in Section 4-4-1, in which the magnetic field in one of the wings of the 3mE hallway is estimated based on $N = 21{,}931$ measurements. In the second and final experiment, presented in Section 4-4-2, the magnetic field of the full 3mE hallway is estimated based on $N = 41{,}383$ magnetic field measurements.

4-4-1 Estimating the Magnetic Field of the Hallway Wing

In the first large-scale experiment, the magnetic field of one of the 3mE hallway wings is estimated based on $N = 21{,}931$ magnetic field measurements along a two-way path in the wing. The area of interest for this experiment is $[-34\,\text{m}, 34\,\text{m}] \times [-5.25\,\text{m}, 5.25\,\text{m}]$ located at a height of 1.0677 m, requiring many inducing points to ensure good accuracy due to its size. The magnetic field is estimated using the SKI framework with cubic convolution interpolation on an inducing point grid of size $400 \times 40 \times 4$, for a total of 64,000 inducing points. The number of inducing points per dimension is chosen such that several inducing points are present per characteristic length scale ℓ, which is the same as in the previous experiment. The other hyperparameters are also the same, given by Table 4-2. These hyperparameters are expected to result in a good estimation of the magnetic field, as the measurement sample rate is the same (240 Hz) and the characteristics of the magnetic field are expected to be similar. For estimation of the magnetic field using the SKI framework, the pivoted Cholesky decomposition rank is set at $l^{(d)} = 5$ along with a residual tolerance of $\mathbf{r}_{i+1}^\top \mathbf{z}_{i+1} = 10^{-8}$ for the preconditioned conjugate gradient method (Algorithm 4). A total of $T = 500$ Lanczos iterations is used for the computation of the predictive variances (Algorithm 6).

The magnetic field is estimated on a two-dimensional test grid of 200×25 points (5,000 total) at a height of 1.0677 m. The magnitude of the magnetic field estimated with both models is shown in Figure 4-10. The transparency of the predicted magnetic field denotes the uncertainty at that location. The figures clearly show that the predictions near the measurements are more certain than those further away, as can be expected. The shared and scalar potential models result in similar estimated magnetic fields with minor differences. Interesting features in the results are the low-value (blue) areas. The large low-value area in the middle occurs due to the presence of lockers in the 3mE hallway wing. The small low-value area at the start of the path is an elevator. The time required to estimate the magnetic field with both models is given in Table 4-5. Due to the large number of test points, the time associated with the computation of the test points is a significant part of the total time. The computation times can be reduced by lowering the number of inducing points, although this will result in a loss of accuracy. Contrarily, the accuracy of the estimated magnetic field can be increased by increasing the number of inducing points at the cost of longer computation times.

Table 4-5: Computation times of the estimation of the magnetic field in the hallway wing.

Model	Train	Test
Shared	31.92 s	3.28 s
Scalar potential	58.46 s	17.48 s

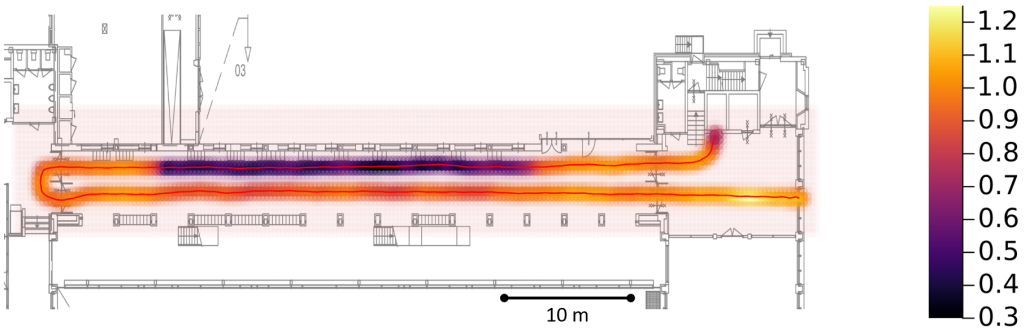
4-4-2 Estimating the Magnetic Field of the Full Hallway

To further emphasize the scalability of modeling magnetic field using SKI for Gaussian process regression, the magnetic field of the full hallway of the 3mE building is estimated based on $N = 41{,}383$ measurements. The data are collected along a one-way weaving path covering an area of interest of $[-100.75\,\text{m}, 100.75\,\text{m}] \times [-4.5\,\text{m}, 4.5\,\text{m}]$ located at a height of 1.0637 m. The same hyperparameters are used for the estimation of the magnetic field again, given by Table 4-2. Also, the same inducing point grid of size $400 \times 40 \times 4$ is used for a total of 64,000 inducing points, ensuring at least one inducing point is present per characteristic length scale ℓ. For estimation of the magnetic field, a residual tolerance of $\mathbf{r}_{i+1}^\top \mathbf{z}_{i+1} = 10^{-8}$ is specified for the preconditioned conjugate gradient algorithm (Algorithm 4) with a pivoted Cholesky decomposition rank of $l^{(d)} = 5$ for $d = 1, 2, 3$, and $T = 500$ Lanczos iterations (Algorithm 6).

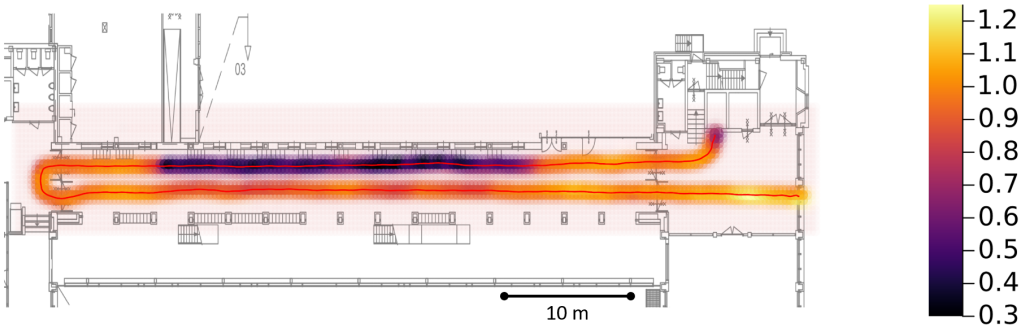
In this experiment, the magnetic field is estimated at a height of 1.0637 m with a two-dimensional test point grid again of size 200×25 for a total of $N_* = 5{,}000$ test points. The magnitude of the estimated magnetic field is shown in Figure 4-11 with the transparency indicating the uncertainty of the prediction at that location. Similar to the previous experiment, the models return the predictions with more certainty near the measurement path. The right wing of the hallway is the area modeled in the previous experiment, where the low-value (blue) areas occur due to the lockers and the elevator. The locations of these low-value areas do not directly coincide with those in the previous experiment due to drift present in the position estimates. The required computation times are given in Table 4-6. For estimation with the shared model, three systems are solved with over 40,000 measurements and 5,000 test points per system in less than one minute. For the scalar potential model, effectively solving a system with over 120,000 measurements and 15,000 test points in less than two minutes.

Table 4-6: Computation times of the estimation of the magnetic field in the full hallway.

Model	Train	Test
Shared	44.77 s	3.05 s
Scalar potential	97.04 s	17.49 s

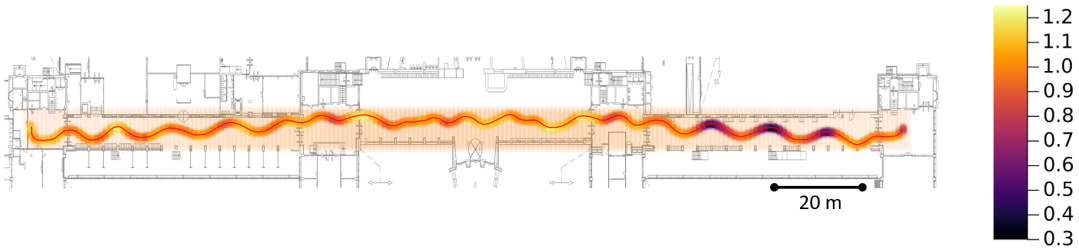


(a) Shared model

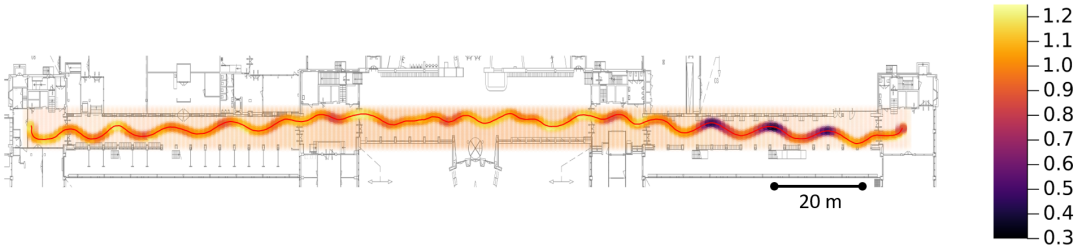


(b) Scalar potential model

Figure 4-10: The magnitude of the predicted magnetic field of the 3mE hallway wing at a height of 1.0677 m with both models. The red line shows the walked path. The transparency of the test values indicates the uncertainty of the predictions.



(a) Shared model



(b) Scalar potential model

Figure 4-11: The magnitude of the predicted magnetic field of the full 3mE hallway at a height of 1.0637 m with both models. The red line shows the walked path. The transparency of the test values indicates the uncertainty of the predictions.

Chapter 5

Conclusions

Indoor positioning systems based on the magnetic field require accurate models of the magnetic field. An established approach for magnetic field modeling is Gaussian process regression, which estimates the magnetic field at unvisited locations based on magnetic field measurements with corresponding positions. Gaussian process regression is used as it is a powerful approach for predicting non-linear data and it is modifiable depending on the application. Additionally, uncertainty information of the predicted magnetic field is provided, making Gaussian process regression suitable for use with (extended) Kalman filters and particle filters, both commonly used in magnetic field localization and SLAM algorithms. A significant drawback of Gaussian process regression is the associated scalability issues, which generally occur when over 10,000 measurements are used on a standard computer. Over the past decades, scalable Gaussian process regression approximations have been researched extensively, of which only a few have been applied in the context of magnetic field modeling: the Laplace operator eigenbasis approximation [14], which uses a limited number of basis functions to approximate the kernel function, and approximations based on local experts [17], estimating the magnetic field using several Gaussian process models on split data sets.

A favorable approach from literature for scalable Gaussian process regression which has not been used for magnetic field modeling is the structured kernel interpolation (SKI) framework. The SKI framework aims to speed up regression through M inducing points, forced into a Cartesian grid structure for kernel interpolation. This grid structure allows for the exploitation of fast matrix-vector multiplications (MVMs) through efficient Krylov subspace methods for fast inference. The SKI framework has a lower computational complexity for magnetic field modeling compared to the Laplace operator eigenbasis approach, reducing the computational complexity from $\mathcal{O}(NM_{\text{bf}}^2)$ to $\mathcal{O}(K(N+3M^{4/3}))$ (for equal numbers of inducing points per dimension; see Section 3-4-1 for further elaboration). $K = J+2T \ll N, M$ is based on the numbers of conjugate gradient iterations J and Lanczos iterations T. However, the accuracy of magnetic field estimation is not directly proportionate between the number of inducing points M and the number of basis functions M_{bf}. Additionally, the SKI framework does not require the prediction to revert to the prior near the boundaries. In this thesis, the SKI framework has been applied to two distinct magnetic field models: the shared model, which

predicts the three components of the magnetic field using three separate Gaussian processes with shared hyperparameters, and the scalar potential model, which follows from Maxwell's equations assuming no magnetization and free currents are present in the domain of interest.

For the first research subquestion, the integration of the SKI framework in the shared and scalar potential models for scalable magnetic field modeling is described in Chapter 3. The squared exponential kernel used for magnetic field modeling is interpolated on the structured inducing points using cubic or quintic convolution interpolation. Additionally, the grid-specific structure in the inducing points together with the squared exponential kernel enable fast MVMs with the inducing point covariance matrix. Through the use of Krylov subspace methods, most notably the preconditioned conjugate gradient method (Algorithm 4, Section 3-4-2) and Lanczos tridiagonalization (Algorithm 6, Section 3-4-3), the fast MVMs are exploited for efficient and scalable inference with the shared and scalar potential models. Within the framework, the number of inducing points can be increased for more accurate results at the cost of computation time.

For the second research subquestion concerning the performance of the SKI framework for scalable magnetic field modeling in terms of accuracy and efficiency, the models were numerically implemented and compared to full Gaussian process regression through simulations and experiments with data collected with the motion capture suit in Sections 4-2 and 4-3. It is shown the use of the SKI framework significantly speeds up magnetic field estimation using Gaussian process regression while retaining accuracy, even for relatively small data sets ($\sim 5.8\times$ and $\sim 11.9\times$ for the shared and scalar potential models respectively with 2,500 three-dimensional measurements at good accuracy levels). The speedups get exponentially larger when more training points are used due to the cubic computational complexity associated with full Gaussian process regression ($\mathcal{O}(N^3)$). For the simulations, both cubic and quintic convolution interpolation were used for kernel interpolation, where quintic convolution interpolation is expected to have a higher accuracy at the cost of computation time when using the same inducing point grid. In the simulations, however, it is shown the use of quintic convolution interpolation does not necessarily lead to better results due to the two extra inducing points required outside the domain of interest on each side per dimension (cubic convolution interpolation has well-defined boundary conditions). For the experiments with magnetic field measurements collected with the motion capture suit, only cubic convolution interpolation was used.

For the final research subquestion concerning the scalability of the SKI framework for magnetic field modeling, experiments were conducted with magnetic field data sets that are too large for full Gaussian process regression in Section 4-4. The first large-scale experiment was conducted with a data set consisting of 21,931 three-dimensional magnetic field measurements. The magnetic field was estimated at 5,000 test locations using 64,000 inducing points. Estimation of the magnetic field only took about 35 seconds with the shared model and roughly 75 seconds with the scalar potential model. To further demonstrate the scalability, a second experiment was conducted using 41,383 three-dimensional magnetic field measurements. Again, the magnetic field was estimated at 5,000 test locations using 64,000 inducing points. In this experiment, estimation of the magnetic field took about 48 seconds with the shared model and roughly 115 seconds with the scalar potential model. While the number of measurements was approximately doubled in the second large-scale experiment, the computation time scaled less due to the split computational complexity (between the number of training and inducing points) of the SKI framework for magnetic field modeling.

To answer the main research question, scalable magnetic field modeling can be achieved within the SKI framework through structured kernel interpolation with cubic convolution interpolation and efficient Krylov subspace methods. Due to the low computational complexity associated with the SKI framework for magnetic field modeling, many training points can be used, more than the computational threshold for full Gaussian process regression. Through experiments, it was demonstrated the framework can be used to construct magnetic field maps based on over 40,000 three-dimensional magnetic field measurements in less than one minute on a standard laptop.

The first direction suggested for further research concerns the numerical implementation of the SKI framework. Algorithms in existing Gaussian process libraries may prove to be useful for speeding up inference. The main library containing the SKI framework is GPyTorch [45]. While this library unfortunately does not contain all algorithms needed for the estimation of the magnetic field with the scalar potential model, implemented algorithms can be studied for faster inference, most notably the modified Batched Conjugate Gradient (mBCG) algorithm. Additionally, by partitioning the MVMs required for the mBCG algorithm, advantage can be taken of parallel computing to increase the number of training points that can be used [50].

The second suggested direction for further research direction concerns the desire to use magnetic field mapping methods and the suit in online settings, such as SLAM. Online mapping requires sequential estimation, which, for magnetic field modeling, is currently possible with the existing Laplace operator eigenbasis approach [14]. This approach has been used in various SLAM applications [8, 18]. Sequential estimation within the SKI framework has already been researched and proposed in existing literature [51], which can be tailored to suit magnetic field modeling using the shared and scalar potential models. In an experimental setup for sequential estimation with suit data, either model may be used, depending on the desired properties of the model.

The final suggested research direction considers the omission of the constant and linear kernel functions in the shared and scalar potential models throughout this thesis, removing the uncertainty estimation of the constant deviation of Earth's magnetic field. These kernel functions were omitted as the linear kernel function is a dot product kernel, which does not lead to any favorable structure in the covariance matrix for fast MVMs. If the linear kernel function could be approximated for favorable structure, fast MVMs can be enabled similarly to fast MVMs with an added constant kernel function as described in Appendix A. This allows for the estimation of the constant deviation of Earth's magnetic field through hyperparameter optimization. Scalable hyperparameter optimization for magnetic field modeling using the SKI framework has not been used in this thesis but is described in Appendix B.

Appendix A

Additive Composite Kernel Matrix-Vector Multiplications

When adding the constant kernel to the squared exponential kernel for estimation of the constant deviation of Earth's magnetic field, the structure of the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ changes. As a result, the composite kernel no longer decomposes as a product kernel over dimensions, supposedly losing the ability to exploit fast MVMs. Fortunately, as summed kernels mean the covariance matrices can simply be summed, fast MVMs exploiting the structure of $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ are still enabled, following the relation

$$\begin{aligned}\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{v}_\mathbf{f} &= \left(\mathbf{K}^{\text{const.}}_{\mathbf{u},\mathbf{u}} + \mathbf{K}^{\text{SE}}_{\mathbf{u},\mathbf{u}}\right)\mathbf{v}_\mathbf{f} \\ &= \mathbf{K}^{\text{const.}}_{\mathbf{u},\mathbf{u}}\mathbf{v}_\mathbf{f} + \mathbf{K}^{\text{SE}}_{\mathbf{u},\mathbf{u}}\mathbf{v}_\mathbf{f}.\end{aligned} \tag{A-1}$$

One would expect twice the number of MVMs to be required for inference with this composite kernel. However, multiplication of the constant covariance matrix $\mathbf{K}^{\text{const.}}_{\mathbf{u},\mathbf{u}}$ with $\mathbf{v}_\mathbf{f} \in \mathbb{R}^M$ is trivial, given $\mathbf{K}^{\text{const.}}_{\mathbf{u},\mathbf{u}}\mathbf{v}_\mathbf{f} = \mathbf{v}_\mathbf{u} \in \mathbb{R}^M$, where $\mathbf{v}_\mathbf{u}$ can be efficiently formulated as $\mathbf{v}_\mathbf{u} = (\sigma_c^2\mathbf{1}^\top\mathbf{v}_\mathbf{f})\mathbf{1}$.

To use the composite kernel with the preconditioned conjugate gradient method, a good preconditioner is required. As mentioned in Section 3-4-2, the preconditioner $\mathbf{P}$ is computed based on the pivoted Cholesky decomposition of $\mathbf{K}_{\mathbf{u},\mathbf{u}}$. For the composite kernel, it is defined by the pivoted Cholesky decompositions of the constant covariance matrix and the squared exponential covariance matrix, i.e.,

$$\begin{aligned}\mathbf{K}_{\mathbf{u},\mathbf{u}} &= \mathbf{K}^{\text{const.}}_{\mathbf{u},\mathbf{u}} + \mathbf{K}^{\text{SE}}_{\mathbf{u},\mathbf{u}} \\ &\approx \mathbf{L}^{\text{const.}}_L \left(\mathbf{L}^{\text{const.}}_L\right)^\top + \mathbf{L}^{\text{SE}}_L \left(\mathbf{L}^{\text{SE}}_L\right)^\top.\end{aligned} \tag{A-2}$$

Both pivoted Cholesky decompositions decompose as Kronecker products similar to equation (3-29). Computation of the pivoted Cholesky decompositions of $\mathbf{L}^{\text{const.}}_L$ and $\mathbf{L}^{\text{SE}}_L$ per dimension is described by Algorithm 5. While for $\mathbf{L}^{\text{SE}}_L$ the pivoted Cholesky decomposition per dimension is computed for the $l^{(d)}$-rank, $\mathbf{L}^{\text{const.}}_L$ only requires the 1-rank pivoted Cholesky

decomposition, given $\mathbf{K}_{\mathbf{u},\mathbf{u}}^{\text{const.}}$ is constant matrix. This allows for a new formulation of the preconditioner $\mathbf{P}$, expressed as

$$\begin{aligned}
\mathbf{P} &= \mathbf{W_f}\left(\mathbf{L}_1^{\text{const.}}\left(\mathbf{L}_1^{\text{const.}}\right)^\top + \mathbf{L}_L^{\text{SE}}\left(\mathbf{L}_L^{\text{SE}}\right)^\top\right)\mathbf{W_f}^\top + \sigma_{\text{n}}^2\mathbf{I} \\
&= \mathbf{W_f}\mathbf{L}_1^{\text{const.}}\left(\mathbf{L}_1^{\text{const.}}\right)^\top\mathbf{W_f}^\top + \mathbf{W_f}\mathbf{L}_L^{\text{SE}}\left(\mathbf{L}_L^{\text{SE}}\right)^\top\mathbf{W_f}^\top + \sigma_{\text{n}}^2\mathbf{I} \\
&= \begin{bmatrix}\mathbf{W_f} & \mathbf{W_f}\end{bmatrix}\begin{bmatrix}\mathbf{L}_1^{\text{const.}} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_L^{\text{SE}}\end{bmatrix}\begin{bmatrix}\left(\mathbf{L}_1^{\text{const.}}\right)^\top & \mathbf{0} \\ \mathbf{0} & \left(\mathbf{L}_L^{\text{SE}}\right)^\top\end{bmatrix}\begin{bmatrix}\mathbf{W_f}^\top \\ \mathbf{W_f}^\top\end{bmatrix} + \sigma_{\text{n}}^2\mathbf{I} \\
&= \overline{\mathbf{W}}_\mathbf{f}\bar{\mathbf{L}}\bar{\mathbf{L}}^\top\overline{\mathbf{W}}_\mathbf{f}^\top + \sigma_{\text{n}}^2\mathbf{I},
\end{aligned} \tag{A-3}$$

where $\overline{\mathbf{W}}_\mathbf{f} \in \mathbb{R}^{N\times 2M}$ and $\bar{\mathbf{L}} \in \mathbb{R}^{2M\times(L+1)}$. Computation of $\mathbf{P}^{-1}\mathbf{r}$ then follows similar to equation (3-31), i.e.,

$$\begin{aligned}
\mathbf{P}^{-1}\mathbf{r} &= \left(\overline{\mathbf{W}}_\mathbf{f}\bar{\mathbf{L}}\bar{\mathbf{L}}^\top\overline{\mathbf{W}}_\mathbf{f}^\top + \sigma_{\text{n}}^2\mathbf{I}\right)^{-1}\mathbf{r} \\
&= \frac{1}{\sigma_{\text{n}}^2}\mathbf{r} - \frac{1}{\sigma_{\text{n}}^4}\overline{\mathbf{W}}_\mathbf{f}\bar{\mathbf{L}}\left(\mathbf{I} + \frac{1}{\sigma_{\text{n}}^2}\bar{\mathbf{L}}^\top\overline{\mathbf{W}}_\mathbf{f}^\top\overline{\mathbf{W}}_\mathbf{f}\bar{\mathbf{L}}\right)^{-1}\bar{\mathbf{L}}^\top\overline{\mathbf{W}}_\mathbf{f}^\top\mathbf{r}.
\end{aligned} \tag{A-4}$$

Solving this system has an associated computational complexity of $\mathcal{O}(2M(L+1)^2)$.

Appendix B

Scalable Hyperparameter Optimization for Magnetic Field Modeling

Similar to the predictive distributions, optimization of the hyperparameters for magnetic field modeling can also be approximated within the SKI framework. This appendix describes hyperparameter optimization using the SKI framework for the shared and scalar potential models. The approach has not been used in the thesis, as it has not been implemented successfully; Julia's optimizers did not reach convergence with the described approach. This appendix provides the necessary theoretical background for future numerical implementations.

Hyperparameter optimization of the shared and scalar potential models in the SKI framework is done by maximizing modified versions of their log marginal likelihood functions, which are given by equations (2-14) and (2-19). The modified log marginal likelihood function for the shared model is given by

$$\begin{aligned}\log p(\mathbf{Y}|\boldsymbol{\theta}) = -\frac{1}{2}\sum_{d=1}^{3}\left(\left(\mathbf{Y}^{(d)}\right)^{\top}\left(\mathbf{W_f}\mathbf{K_{u,u}}\mathbf{W_f}^{\top}+\sigma_{\mathrm{n}}^{2}\mathbf{I}\right)^{-1}\mathbf{Y}^{(d)}\right) - \\ \frac{3}{2}\log\left|\mathbf{W_f}\mathbf{K_{u,u}}\mathbf{W_f}^{\top}+\sigma_{\mathrm{n}}^{2}\mathbf{I}\right| - \frac{3N}{2}\log 2\pi,\end{aligned} \tag{B-1}$$

where $\mathbf{A} = \mathbf{W_f}\mathbf{K_{u,u}}\mathbf{W_f}^{\top}+\sigma_{\mathrm{n}}^{2}\mathbf{I}$. For the scalar potential model, the differentiated interpolation scheme of equation (3-7) is used, resulting in the log marginal likelihood function given by

$$\begin{aligned}\log p(\mathbf{Y}|\boldsymbol{\theta}) = -\frac{1}{2}\operatorname{vec}\left(\mathbf{Y}^{\top}\right)^{\top}\left(\left(\partial\mathbf{W_f}\right)\mathbf{K_{u,u}}\left(\partial\mathbf{W_f}\right)^{\top}+\sigma_{\mathrm{n}}^{2}\mathbf{I}\right)^{-1}\operatorname{vec}\left(\mathbf{Y}^{\top}\right) - \\ \frac{1}{2}\log\left|\left(\partial\mathbf{W_f}\right)\mathbf{K_{u,u}}\left(\partial\mathbf{W_f}\right)^{\top}+\sigma_{\mathrm{n}}^{2}\mathbf{I}\right| - \frac{N}{2}\log 2\pi,\end{aligned} \tag{B-2}$$

where $\partial\mathbf{A} = (\partial\mathbf{W_f})\mathbf{K_{u,u}}(\partial\mathbf{W_f})^{\top}+\sigma_{\mathrm{n}}^{2}\mathbf{I}$.

Fortunately, the modified log marginal likelihood functions can be computed efficiently within the SKI framework. Computation of the data fit requires the same approach as the solution to the linear system of equation (3-24). The optimal solution to the linear system can be found

using the preconditioned conjugate gradient method and only needs to be left-multiplied by $(\mathbf{Y}^{(d)})^\top$ or $\text{vec}(\mathbf{Y}^\top)^\top$ for the shared and scalar potential models respectively to find the data fit. The computation of the complexity penalty is slightly more complicated. Its associated computational complexity is based on the computation of the log determinant of a matrix of size $N \times N$ for the shared model and $3N \times 3N$ for the scalar potential model. The standard procedure for full Gaussian process regression is to use the Cholesky decomposition, which is undesirable due to the associated cubic computational complexity [19]. A scalable approach for estimation of the log determinant in the SKI framework is based on stochastic trace estimators, exploiting fast MVMs through the Lanczos tridiagonalization algorithm [52].

This approach relies on the equality between the log determinant and the trace of the matrix logarithm, expressed as

$$\log|\mathbf{A}| = \text{tr}(\log(\mathbf{A})) \tag{B-3}$$

The trace of $\log(\mathbf{A})$ can also be expressed as

$$\text{tr}(\log(\mathbf{A})) = \mathbb{E}\left[\mathbf{z}^\top \log(\mathbf{A})\mathbf{z}\right], \tag{B-4}$$

where $\mathbf{z} \in \mathbb{R}^N$ is a random probe vector with zero mean and variance one, often chosen to be vectors with random Rademacher variables as its entries, i.e., $z_i \in \{-1, 1\}$ for $i = 1, 2, \ldots, N$ [52]. The trace is then stochastically estimated as the mean over N_z random probe vectors. Estimation of $\log|\mathbf{A}|$ is done using the connection between the Gaussian quadrature rule and the Lanczos algorithm [52], and is described by Algorithm 7 [53]. Relatively few probe vectors and Lanczos iterations are required for a good estimation, where just 5 probe vectors are needed with 25 Lanczos iterations each for a good approximation on almost 60,000 training points [52]. As the scalable log determinant approach is based on fast MVMs, its use fits right into the SKI framework. Using the sparse interpolation matrix $\mathbf{W_f}$ and the Kronecker structure in $\mathbf{K_{u,u}}$ for fast MVMs, the log determinant can be computed efficiently. Again, as was the case for the computation of the predictive distribution of the scalar potential model, the log determinant for the scalar potential model can be approximated by substituting $\partial\mathbf{A}$ for $\mathbf{A}$ and $\partial\mathbf{W_f}$ for $\mathbf{W_f}$.

In numerical implementations, gradients are often specified so efficient gradient-based optimization algorithms can be used for faster convergence to the optimum. The log marginal likelihood functions for the shared and scalar potential models in the SKI framework of equations (B-1) and (B-2) can be differentiated with respect to the hyperparameters $\boldsymbol{\theta}$. These differentiated log marginal likelihood functions for the shared and scalar potential models are respectively given by

$$\frac{\partial}{\partial\boldsymbol{\theta}_k}\log p(\mathbf{Y}|\boldsymbol{\theta}) = \frac{1}{2}\sum_{d=1}^{3}\left(\left(\mathbf{Y}^{(d)}\right)^\top \mathbf{A}^{-1}\frac{\partial(\mathbf{A})}{\partial\boldsymbol{\theta}_k}\mathbf{A}^{-1}\mathbf{Y}^{(d)}\right) - \frac{3}{2}\,\text{tr}\left(\mathbf{A}^{-1}\frac{\partial(\mathbf{A})}{\partial\boldsymbol{\theta}_k}\right), \tag{B-5}$$

$$\frac{\partial}{\partial\boldsymbol{\theta}_k}\log p(\mathbf{Y}|\boldsymbol{\theta}) = \frac{1}{2}\,\text{vec}\left(\mathbf{Y}^\top\right)^\top (\partial\mathbf{A})^{-1}\frac{\partial(\partial\mathbf{A})}{\partial\boldsymbol{\theta}_k}(\partial\mathbf{A})^{-1}\,\text{vec}\left(\mathbf{Y}^\top\right) - \frac{1}{2}\,\text{tr}\left((\partial\mathbf{A})^{-1}\frac{\partial(\partial\mathbf{A})}{\partial\boldsymbol{\theta}_k}\right), \tag{B-6}$$

which are differentiated using the *derivative of inverse* and *derivative of log determinant* relations [10]. These relations are given by

$$\frac{\partial}{\partial\boldsymbol{\theta}_k}\mathbf{A}^{-1} = -\mathbf{A}^{-1}\frac{\partial(\mathbf{A})}{\partial\boldsymbol{\theta}_k}\mathbf{A}^{-1}, \tag{B-7a}$$

$$\frac{\partial}{\partial \boldsymbol{\theta}_k} \log |\mathbf{A}| = \operatorname{tr}\left(\mathbf{A}^{-1} \frac{\partial(\mathbf{A})}{\partial \boldsymbol{\theta}_k}\right). \tag{B-7b}$$

The matrix $\frac{\partial(\mathbf{A})}{\partial \boldsymbol{\theta}_k}$ is computed elementwise using the derivatives of the kernel function. The derivatives of the squared exponential kernel used for magnetic field modeling are given by

$$\frac{\partial}{\partial \ell} \kappa_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = \sigma_{\mathrm{f}}^2 \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\ell^2}\right) \frac{||\mathbf{x} - \mathbf{x}'||^2}{\ell^3}, \tag{B-8a}$$

$$\frac{\partial}{\partial \sigma_{\mathrm{f}}} \kappa_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = 2\sigma_{\mathrm{f}} \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\ell^2}\right), \tag{B-8b}$$

The derivative of the covariance matrix with respect to the noise hyperparameter is given by

$$\frac{\partial(\mathbf{A})}{\partial \sigma_{\mathrm{n}}} = 2\sigma_{\mathrm{n}}\mathbf{I}. \tag{B-9}$$

For the efficient computation of the gradients of the log marginal likelihood functions, the only additional requirements are the computation of the gradient of the log determinant and the specification of an MVM approach with each $\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}_k}$ (or $\frac{\partial(\partial \mathbf{A})}{\partial \boldsymbol{\theta}_k}$).

For fast MVMs with each $\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}_k}$ (or $\frac{\partial(\partial \mathbf{A})}{\partial \boldsymbol{\theta}_k}$, the MVM approach is based on the structure in the differentiated covariance matrix $\frac{\partial}{\partial \boldsymbol{\theta}_k}\mathbf{K}_{\mathbf{u},\mathbf{u}}$. Fast MVMs involving the derivative with respect to the noise hyperparameter are trivial, given the differentiated covariance matrix is diagonal. For the signal variance σ_{f}, fast MVMs can be achieved as described in Section 3-4-1 with the modified kernel function of equation (B-8b). Fast MVMs involving the characteristic length scale ℓ derivative are less straightforward. The kernel function of equation (B-8a) cannot be expressed as a product over grid dimensions. Fortunately, a favorable structure in the differentiated covariance matrix is still present as the kernel function is stationary ($\kappa(\mathbf{x}, \mathbf{x}') = \kappa(\mathbf{x} - \mathbf{x}')$). This results in a symmetric block-Toeplitz with Toeplitz blocks (BTTB) matrix [40]. Toeplitz matrices allow for fast MVMs via circulant embeddings, which are computationally attractive as their eigendecomposition is efficiently computed with the fast Fourier transform (FFT). This approach can be extended to BTTB matrices using the multi-dimensional FFT. For further elaboration, see the paper by Wilson et al. [40].

The gradient of the log determinant for the complexity penalty can be computed as a byproduct of the Lanczos tridiagonalization algorithm executed for the non-differentiated log determinant (see Algorithm 7). The differentiated log determinant can then also be stochastically estimated [52], which can be efficiently computed using the fast MVMs described in the previous paragraph through the expression

$$\begin{aligned} \operatorname{tr}\left(\mathbf{A}^{-1} \frac{\partial(\mathbf{A})}{\partial \boldsymbol{\theta}_k}\right) &= \mathbb{E}\left[\left(\mathbf{A}^{-1}\mathbf{z}\right)^\top \left(\frac{\partial(\mathbf{A})}{\partial \boldsymbol{\theta}_k}\right) \mathbf{z}\right] \\ &\approx \mathbb{E}\left[\left(\mathbf{Q}_k \left(\mathbf{T}_k^{-1} \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}^\top ||\mathbf{z}||\right)\right)^\top \left(\frac{\partial(\mathbf{A})}{\partial \boldsymbol{\theta}_k}\right) \mathbf{z}\right], \end{aligned} \tag{B-10}$$

Fast computation of the data fit, the complexity penalty, and their derivatives allows for scalable hyperparameter optimization for magnetic field modeling using efficient gradient-based optimization algorithms, such as the (L-)BFGS algorithm. Many magnetic field measurements can be used, even above the prohibitive threshold for full Gaussian process regression

(approximately $10,000$ and $3,300$ for the shared and scalar potential models respectively). The optimization parameters are $\boldsymbol{\theta} = [\ell \quad \sigma_\mathrm{f} \quad \sigma_\mathrm{n}]^\top$, as the constant and linear kernels are omitted due to the lack of favorable structure introduced by the linear kernel (see Section 3-4-1). In numerical implementations, their exponentiated versions are used as described by equation (2-7) to ensure the found hyperparameters are positive without constraining the optimization problem.

Algorithm 7 Estimation of the matrix log determinant $\log|\mathbf{A}|$ [53]

Input: $\mathbf{K}_{\mathbf{u},\mathbf{u}} \in \mathbb{R}^{m^{(d)} \times m^{(d)}}$ for $d = 1$ to D, $\mathbf{W}_\mathbf{f} \in \mathbb{R}^{N \times M}$, σ_n, N_z
Output: approximated $\log|\mathbf{A}|$

for $k = 1$ to N_z **do**
    generate random Rademacher vector $\mathbf{z}_k \in \mathbb{R}^N$
    normalize $\mathbf{z}_k$ to find probe vector $\mathbf{z}_\mathrm{probe}$
    compute $\mathbf{T}_T$ using the Lanczos algorithm with probe vector $\mathbf{z}_\mathrm{probe}$ (Algorithm 6)
    compute the eigenvalues $\boldsymbol{\lambda}$ and the eigenvectors $\mathbf{e}$ of $\mathbf{T}_T$
    define τ_t as the first element of the tth eigenvector $\mathbf{e}_t$
    $\mathbf{z}_k \log(\mathbf{A})\mathbf{z}_k \approx \sum_{t=1}^{T} \tau_t^2 \log(\lambda_t)$
end for
$\log|\mathbf{A}| \approx \frac{N}{N_z} \sum_{k=1}^{N_z} \mathbf{z}_k \log(\mathbf{A})\mathbf{z}_k$

Bibliography

[1] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019.

[2] H. Obeidat, W. Shuaieb, O. Obeidat, and R. Abd-Alhameed, "A review of indoor localization techniques and wireless technologies," *Wireless Personal Communications*, vol. 119, no. 1, pp. 289–327, 2021.

[3] J. D. Hol, *Sensor fusion and calibration of inertial sensors, vision, ultra-wideband and GPS*. PhD thesis, Linköping University Electronic Press, 2011.

[4] Apple, "Location services & privacy," Sep 2022. [Online; accessed December 22, 2022], https://www.apple.com/legal/privacy/data/en/location-services.

[5] B. Li, T. Gallagher, A. G. Dempster, and C. Rizos, "How feasible is the use of magnetic field alone for indoor positioning?," in *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–9, IEEE, 2012.

[6] S. Park and S. Hashimoto, "Indoor localization for autonomous mobile robot based on passive RFID," in *2008 IEEE international conference on robotics and biomimetics*, pp. 1856–1861, IEEE, 2009.

[7] J. Haverinen and A. Kemppainen, "Global indoor self-localization based on the ambient magnetic field," *Robotics and Autonomous Systems*, vol. 57, no. 10, pp. 1028–1035, 2009.

[8] F. Viset, R. Helmons, and M. Kok, "An extended Kalman filter for magnetic field SLAM using Gaussian process regression," *Sensors*, vol. 22, no. 8, p. 2833, 2022.

[9] A. Solin, S. Särkkä, J. Kannala, and E. Rahtu, "Terrain navigation in the magnetic landscape: Particle filtering for indoor positioning," in *2016 European Navigation Conference (ENC)*, pp. 1–9, IEEE, 2016.

[10] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2. MIT press Cambridge, MA, 2006.

[11] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning, "Simultaneous localization and mapping using ambient magnetic field," in *2010 IEEE Conference on Multisensor Fusion and Integration*, pp. 14–19, IEEE, 2010.

[12] N. Wahlström, M. Kok, T. B. Schön, and F. Gustafsson, "Modeling magnetic fields using Gaussian processes," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3522–3526, IEEE, 2013.

[13] N. Wahlström, *Modeling of magnetic fields and extended objects for localization applications*. PhD thesis, Linköping University Electronic Press, 2015.

[14] A. Solin, M. Kok, N. Wahlström, T. B. Schön, and S. Särkkä, "Modeling and interpolation of the ambient magnetic field by Gaussian processes," *IEEE Transactions on robotics*, vol. 34, no. 4, pp. 1112–1127, 2018.

[15] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When Gaussian process meets big data: A review of scalable GPs," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 11, pp. 4405–4423, 2020.

[16] A. Solin and S. Särkkä, "Hilbert space methods for reduced-rank Gaussian process regression," *Statistics and Computing*, vol. 30, no. 2, pp. 419–446, 2020.

[17] M. Kok and A. Solin, "Scalable magnetic field SLAM in 3D using Gaussian process maps," in *2018 21st international conference on information fusion (FUSION)*, pp. 1353–1360, IEEE, 2018.

[18] F. Viset, J. T. Gravdahl, and M. Kok, "Magnetic field norm SLAM using Gaussian process regression in foot-mounted sensors," in *2021 European Control Conference (ECC)*, pp. 392–398, IEEE, 2021.

[19] A. Wilson and H. Nickisch, "Kernel interpolation for scalable structured Gaussian processes (KISS-GP)," in *International conference on machine learning*, pp. 1775–1784, PMLR, 2015.

[20] G. Pleiss, J. Gardner, K. Weinberger, and A. G. Wilson, "Constant-time predictive distributions for Gaussian processes," in *International Conference on Machine Learning*, pp. 4114–4123, PMLR, 2018.

[21] T. Veen, "Magnetic field SLAM: using an inertial human motion suit and reduced rank Gaussian process regression," Master's thesis, Delft University of Technology, 2022.

[22] A. Kemppainen, J. Haverinen, I. Vallivaara, and J. Röning, "Near-optimal exploration in Gaussian process SLAM: Scalable optimality factor and model quality rating.," in *ECMR*, pp. 283–290, 2011.

[23] S. Särkkä, "Linear operators and stochastic partial differential equations in Gaussian process regression," in *International Conference on Artificial Neural Networks*, pp. 151–158, Springer, 2011.

[24] Xsens, "Interpreting magnetic field data represented as an arbitrary unit (a.u.)," *Xsens knowledge base*, Aug 2022. [Online; accessed November 14,

2022], https://base.xsens.com/s/article/Interpreting-magnetic-field-data-represented-as-an-arbitrary-unit-a-u.

[25] National Geophysical Data Center, "Magnetic field calculators." [Online; accessed December 27, 2022], https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml#igrfwmm.

[26] Xsens Technologies B.V., "MVN Link Suit." [Online; accessed May 10, 2022], https://motionshop.xsens.com/index.php/products/motion-capture/accessories/mvnl-suit.

[27] M. Schepers, M. Giuberti, G. Bellusci, *et al.*, "Xsens MVN: Consistent tracking of human motion using inertial sensing," *Xsens Technol*, vol. 1, no. 8, 2018.

[28] U. Myn, M. Link, and M. Awinda, *Xsens MVN User Manual*, 2015.

[29] N. Richardson, I. Cook, N. Crane, D. Dunnington, R. François, J. Keane, D. Moldovan-Grünfeld, J. Ooms, and Apache Arrow, *arrow: Integration to 'Apache' 'Arrow'*, 2022. https://github.com/apache/arrow/, https://arrow.apache.org/docs/r/.

[30] J. Quinonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *The Journal of Machine Learning Research*, vol. 6, pp. 1939–1959, 2005.

[31] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," *Advances in neural information processing systems*, vol. 13, 2000.

[32] A. Smola and P. Bartlett, "Sparse greedy Gaussian process regression," *Advances in neural information processing systems*, vol. 13, 2000.

[33] D. Eriksson, K. Dong, E. Lee, D. Bindel, and A. G. Wilson, "Scaling Gaussian process regression with derivatives," *Advances in neural information processing systems*, vol. 31, 2018.

[34] P. Izmailov, A. Novikov, and D. Kropotov, "Scalable Gaussian processes with billions of inducing inputs via tensor train decomposition," in *International Conference on Artificial Intelligence and Statistics*, pp. 726–735, PMLR, 2018.

[35] R. J. I. Marks, *Introduction to Shannon sampling and interpolation theory*. Springer Science & Business Media, 2012.

[36] E. H. Meijering, K. J. Zuiderveld, and M. A. Viergever, "Image reconstruction by convolution with symmetrical piecewise nth-order polynomial kernels," *IEEE transactions on image processing*, vol. 8, no. 2, pp. 192–201, 1999.

[37] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE transactions on acoustics, speech, and signal processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

[38] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.

[39] Y. Saatçi, *Scalable inference for structured Gaussian process models*. PhD thesis, Citeseer, 2012.

[40] A. G. Wilson, C. Dann, and H. Nickisch, "Thoughts on massively scalable Gaussian processes," *arXiv preprint arXiv:1511.01870*, 2015.

[41] A. G. Wilson, *Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes*. PhD thesis, Citeseer, 2014.

[42] J. Nocedal and S. J. Wright, "Conjugate gradient methods," *Numerical optimization*, pp. 101–134, 2006.

[43] H. Harbrecht, M. Peters, and R. Schneider, "On the low-rank approximation by the pivoted Cholesky decomposition," *Applied numerical mathematics*, vol. 62, no. 4, pp. 428–440, 2012.

[44] K. Schacke, "On the Kronecker product," *Master's thesis, University of Waterloo*, 2004.

[45] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, "GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," *Advances in neural information processing systems*, vol. 31, 2018.

[46] G. Papandreou and A. L. Yuille, "Efficient variational inference in large-scale Bayesian compressed sensing," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1332–1339, IEEE, 2011.

[47] J. W. Demmel, *Applied numerical linear algebra*. SIAM, 1997.

[48] P. Arbenz, "Lecture notes on solving large scale eigenvalue problems," 2016.

[49] M. Kok, J. D. Hol, and T. B. Schön, "Using inertial sensors for position and orientation estimation," *arXiv preprint arXiv:1704.06053*, 2017.

[50] K. Wang, G. Pleiss, J. Gardner, S. Tyree, K. Q. Weinberger, and A. G. Wilson, "Exact Gaussian processes on a million data points," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[51] S. Stanton, W. Maddox, I. Delbridge, and A. G. Wilson, "Kernel interpolation for scalable online Gaussian processes," in *International Conference on Artificial Intelligence and Statistics*, pp. 3133–3141, PMLR, 2021.

[52] K. Dong, D. Eriksson, H. Nickisch, D. Bindel, and A. G. Wilson, "Scalable log determinants for Gaussian process kernel learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[53] S. Ubaru, J. Chen, and Y. Saad, "Fast estimation of tr(f(A)) via stochastic Lanczos quadrature," *SIAM Journal on Matrix Analysis and Applications*, vol. 38, no. 4, pp. 1075–1099, 2017.

Glossary

List of Acronyms

BTTB	Block-Toeplitz With Toeplitz Blocks
DCSC	Delft Center for Systems and Control
D-SKI	Structured Kernel Interpolation With Derivatives
FFT	Fast Fourier Transform
FITC	Fully Independent Training Conditional
GPS	Global Positioning System
IMU	Inertial Measurement Unit
(L-)BFGS	(Limited-Memory) Broyden-Fletcher-Goldfarb-Shanno
LOVE	LanczOs Variance Estimates
mBCG	modified Batched Conjugate Gradient
MSLL	Mean Standardized Log Loss
MVM	Matrix-Vector Multiplication
MVNX	MVN Open XML Format
RE	Relative Error
RMSE	Root-Mean-Square Error
SKI	Structured Kernel Interpolation
SLAM	Simultaneous Localization and Mapping
SoR	Subset of Regressors