Journal of Machine Learning Research 27 (2026) 1-60 Submitted 7/25; Revised 5/26; Published 6/26

Vecchia-Inducing-Points Full-Scale Approximations for Gaussian Processes

Tim Gyger TIM.GYGER@HSLU.CH
Lucerne University of Applied Sciences and Arts
Department of Mathematical Modeling and Machine Learning
University of Zurich
Suurstoffi 1, 6343 Rotkreuz, Switzerland

Reinhard Furrer REINHARD.FURRER@UZH.CH
Department of Mathematical Modeling and Machine Learning
University of Zurich

Fabio Sigrist FABIO.SIGRIST@STAT.MATH.ETHZ.CH
Seminar for Statistics, ETH Zurich
Lucerne University of Applied Sciences and Arts

Editor: Chris Oates

Abstract

Gaussian processes are flexible, probabilistic, non-parametric models widely used in machine learning and statistics. However, their scalability to large data sets is limited by computational constraints. To overcome these challenges, we propose **V**ecchia-**i**nducing-points **f**ull-scale (VIF) approximations combining the strengths of global inducing points and local Vecchia approximations. Vecchia approximations excel in settings with low-dimensional inputs and moderately smooth covariance functions, while inducing point methods are better suited to high-dimensional inputs and smoother covariance functions. Our VIF approach bridges these two regimes by using an efficient correlation-based neighbor-finding strategy for the Vecchia approximation of the residual process, implemented via a modified cover tree algorithm. We further extend our framework to non-Gaussian likelihoods by introducing iterative methods that substantially reduce computational costs for training and prediction by several orders of magnitude compared to Cholesky-based computations when using a Laplace approximation. In particular, we propose and compare novel preconditioners and provide theoretical convergence results. Extensive numerical experiments on simulated and real-world data sets show that VIF approximations are both computationally efficient as well as more accurate and numerically stable than state-of-the-art alternatives. All methods are implemented in the open-source C++ library `GPBoost` with high-level Python and R interfaces.

Keywords: Probabilistic modeling, uncertainty quantification, large data, iterative methods, Laplace approximation

1. Introduction

Gaussian process (GP) models are probabilistic non-parametric models which are used across various disciplines including machine learning (Rasmussen and Williams, 2006) and geostatistics (Cressie, 1993). However, their applicability to large data sets is limited by

computational costs, with $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ memory complexity, where n is the number of observations. Various strategies have been proposed to mitigate this computational burden; see Liu et al. (2020) and Heaton et al. (2019) for reviews. Low-rank inducing-points approximations such as the fully independent training conditional (FITC) approximation (Quinonero-Candela and Rasmussen, 2005), predictive process models (Banerjee et al., 2008; Finley et al., 2009), and variational approximations (Titsias, 2009; Hensman et al., 2013, 2015) are global approximations that focus on capturing the large-scale structures of GPs. Such inducing point methods are widely used in machine learning and are considered state-of-the-art methods. An alternative approach to improving computational efficiency is to leverage sparse linear algebra methods by enforcing sparsity in a Cholesky factor of precision matrices (Vecchia, 1988; Datta et al., 2016; Katzfuss and Guinness, 2021; Schäfer et al., 2021a; Cao et al., 2023). These local Vecchia approximations are considered as a "leader among the sea of approximation" (Guinness, 2021) in spatial statistics as they often yield superior approximation accuracy (Rambelli and Sigrist, 2026).

In this work, we propose an approach that integrates inducing point methods with Vecchia approximations and denote this as Vecchia-inducing-points full-scale (VIF) approximations. This method combines the advantages of global inducing points and local Vecchia approximations. Vecchia approximations excel in settings with low-dimensional inputs and moderately smooth covariance functions. The latter can be explained by the screening effect (Schäfer et al., 2021a, Section 3.3.3). Vecchia approximations are expected to work less well for high-dimensional data as they depend on the selection of neighbors, and this is more difficult in higher dimensions. Low-rank inducing point methods, on the other hand, are expected to work well for smoother covariance functions (Schäfer et al., 2021b, Section 2.4) and are better suited to high-dimensional inputs (see our experimental results in Section 7.1). VIF approximations bridge these two types of approximations. In addition, we propose a novel, efficient correlation-distance-based method for selecting conditioning sets for Vecchia approximations on the residual process using a modified cover tree algorithm.

We further extend our framework to non-Gaussian likelihoods to include, e.g., classification and count data regression tasks, by using a Laplace approximation. Although, depending on the likelihood, Laplace approximations can be inaccurate for small data sets, they are computationally very efficient (Nickisch and Rasmussen, 2008) and converge asymptotically to the correct quantity and are thus accurate for large data sets. To support this argument, Figure 1 shows the estimated marginal variance parameter obtained with a VIF-Laplace approximation for varying sample sizes n, based on simulated data with a Bernoulli likelihood as described in Section 7. Estimation is performed using the iterative methods introduced in this paper and repeated across 100 simulated data sets for each n. The results demonstrate that the downward bias in the variance parameter diminishes as n increases. However, standard Cholesky-based calculations do not scale well despite the use of VIF and Laplace approximations for non-Gaussian likelihoods since this requires computing the Cholesky factorization of a sparse $n \times n$ matrix (see Section 3) whose computational complexity is not linear in n and depends on the graph structure as well as the effectiveness of the fill-reducing ordering (Davis, 2006). For two-dimensional spatial data, this complexity is approximately $\mathcal{O}(n^{3/2})$, while in higher dimensions, it is at least $\mathcal{O}(n^2)$ (Lipton et al., 1979). To remedy this, we introduce iterative methods (Saad, 2003; Trefethen and Bau, 2022; Aune et al., 2014; Gardner et al., 2018a) that substantially reduce computational

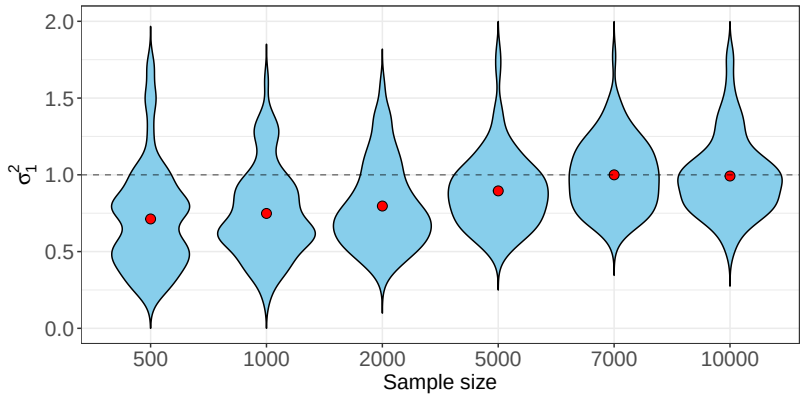

Figure 1: Violin plots of the estimated variance parameter σ_1^2 for different sample sizes n for binary data. Red dots indicate the mean estimates, and the dashed line marks the true parameter value $\sigma_1^2 = 1$.

costs for likelihood evaluations, gradient computations, and the calculation of predictive distributions when using a Laplace approximation. In contrast to the Cholesky decomposition, iterative methods require only matrix-vector multiplications which can be trivially parallelized. We propose and compare novel preconditioners, derive new convergence guarantees, introduce two simulation-based approaches to compute predictive variances, and provide both theoretical insights and empirical evaluations. Furthermore, we compare the prediction accuracy and runtime of our methods against existing state-of-the-art approaches on a large set of simulated and real-world data sets and find that VIF approximations are both computationally efficient and more accurate than state-of-the-art alternatives.

Full-scale approximations were first introduced in Sang et al. (2011) and Sang and Huang (2012), which combined inducing points with covariance tapering (Furrer et al., 2006) approximations. Zhang et al. (2019) extended this by partitioning the data into blocks and deriving a block-wise sparse precision matrix for the residual process. While this can also be interpreted as a special form of a Vecchia approximation, Zhang et al. (2019) do not explicitly consider the case when using single points as conditioning sets, they do not address the question of efficiently selecting neighbors, as this might be less of an issue when using blocks, and they do not cover non-Gaussian likelihoods, which entail additional computational challenges.

The remainder of the paper is structured as follows. Sections 2 and 3 present the VIF framework for Gaussian and non-Gaussian likelihoods, respectively. In Section 4, we introduce iterative methods for inference within the VIF framework, and in Section 5, we present convergence results for the preconditioned conjugate gradient (CG) methods. In Section 6, we propose efficient algorithms based on cover trees to identify the nearest Vecchia neighbors with respect to the correlation distance. In Section 7, we conduct simulation studies to analyze computational and statistical properties of VIF approximations. Finally, in Section 8, we apply our methodology to various real-world data sets, comparing runtime and prediction accuracy with several state-of-the-art methods. Additionally, we extend these

experiments by estimating the smoothness parameter of automatic relevance determination (ARD) Matérn kernels, contributing to the challenge of model selection - at least within the family of Matérn kernels.

1.1 Software implementation

The methods presented in this article are implemented in the `GPBoost` library written in C++ with Python and R interface packages, see `https://github.com/fabsig/GPBoost`.[1] In addition, to facilitate the practical use of the proposed methodology, we provide a self-contained generic Python example in the GitHub repository `https://github.com/TimGyger/VIF` that demonstrates how to fit a VIF GP model and obtain predictions using the `GPBoost` Python interface.

2. VIF approximations for Gaussian process regression

We consider the following GP model:

$$y = F(\boldsymbol{x}) + b(\boldsymbol{s}) + \epsilon(\boldsymbol{s}), \tag{1}$$

where $y \in \mathbb{R}$ is the response variable, $b(\boldsymbol{s})$ is a zero-mean GP with inputs $\boldsymbol{s} \in \mathbb{R}^d$, specified by a parametric covariance function $c_{\boldsymbol{\theta}}(\cdot,\cdot)$ with parameters $\boldsymbol{\theta} \in \mathbb{R}^q$, $F() : \mathbb{R}^p \to \mathbb{R}$ is a fixed effects prior mean function with predictor variables $\boldsymbol{x} \in \mathbb{R}^p$, and $\epsilon(\boldsymbol{s})$ is a zero-mean independent Gaussian white noise process with variance σ^2. Note that the GP inputs $\boldsymbol{s}$ and the fixed effects predictor variables $\boldsymbol{x}$ may or may not be overlapping. For instance, in spatial statistics, $\boldsymbol{s}$ often consists of spatial and/or temporal coordinates whereas $\boldsymbol{x}$ contains additional predictor variables. But in machine learning applications, the GP and fixed effects inputs can also be equal; see, e.g., Sigrist (2022a) for an example. For notational simplicity, we will assume $F(\boldsymbol{x}) = 0$ in the following, but $F(\cdot)$ can be easily extended to a linear mean function $F(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{\beta}$, $\boldsymbol{\beta} \in \mathbb{R}^p$, and to machine learning methods such as tree-boosting (Sigrist, 2022a,b) and neural networks (Simchoni and Rosset, 2023).

If n samples $\boldsymbol{y} = (y(\boldsymbol{s}_1), \dots, y(\boldsymbol{s}_n))^{\mathrm{T}} \in \mathbb{R}^n$ are observed at inputs $\mathcal{S} = \{\boldsymbol{s}_1, \dots, \boldsymbol{s}_n\}$, $\boldsymbol{y}$ has a multivariate distribution with mean zero and covariance matrix

$$\widetilde{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma} + \sigma^2 \boldsymbol{I}_n, \tag{2}$$

where $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$ is the covariance matrix of $\boldsymbol{b} = (b(\boldsymbol{s}_1), \dots, b(\boldsymbol{s}_n))^{\mathrm{T}} \in \mathbb{R}^n$ with entries $\Sigma_{ij} = \mathrm{Cov}\left(b(\boldsymbol{s}_i), b(\boldsymbol{s}_j)\right) = c_{\boldsymbol{\theta}}\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right), \boldsymbol{s}_i, \boldsymbol{s}_j \in \mathcal{S}$. Concerning the notation in this article, we use the symbol $\widetilde{}$ to distinguish matrices including the error variance $\sigma^2 \boldsymbol{I}_n$ from those without this diagonal error variance matrix. E.g., $\widetilde{\boldsymbol{\Sigma}}$ contains the error variance $\sigma^2 \boldsymbol{I}_n$ whereas $\boldsymbol{\Sigma}$ does not. For maximum likelihood estimation, we minimize the negative log-likelihood function

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{y}) = \frac{n}{2}\log(2\pi) + \frac{1}{2}\log\det\left(\widetilde{\boldsymbol{\Sigma}}\right) + \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\widetilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{y}.$$

1. The VIF approximation can be enabled via the parameter `gp_approx = "vif"`, iterative methods can be enabled via the parameter `matrix_inversion_method = "iterative"` (=default), and the preconditioner is chosen via the parameter `cg_preconditioner_type`.

Evaluation of the log-likelihood and its derivatives involves the calculation of linear solves with the $n \times n$ matrix $\widetilde{\boldsymbol{\Sigma}}$ and calculating its determinant, which has $\mathcal{O}\left(n^3\right)$ complexity when using a Cholesky decomposition.

For prediction at n_p new locations $\mathcal{S}^p = \{\boldsymbol{s}_1^p, ..., \boldsymbol{s}_{n_p}^p\}$, the predictive distribution $\boldsymbol{y}^p|\boldsymbol{y}$ is given by $\mathcal{N}(\boldsymbol{\mu}^p, \boldsymbol{\Sigma}^p)$, where $\boldsymbol{\mu}^p = \boldsymbol{\Sigma}_{nn_p}^{\mathrm{T}} \widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{y}$, $\boldsymbol{\Sigma}^p = \boldsymbol{\Sigma}_{n_p} + \sigma^2 \boldsymbol{I}_{n_p} - \boldsymbol{\Sigma}_{nn_p}^{\mathrm{T}} \widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma}_{nn_p} \in \mathbb{R}^{n_p \times n_p}$, $\boldsymbol{\Sigma}_{nn_p} = \left[c_{\boldsymbol{\theta}}\left(\boldsymbol{s}_i, \boldsymbol{s}_j^p\right)\right]_{i=1:n, j=1:n_p}$ is a cross-covariance matrix, and $\boldsymbol{\Sigma}_{n_p} = \left[c_{\boldsymbol{\theta}}\left(\boldsymbol{s}_i^p, \boldsymbol{s}_j^p\right)\right]_{i,j=1:n_p}$. For large n_p, it is usually infeasible to store the complete predictive covariance matrix $\boldsymbol{\Sigma}^p$, and the primary focus is solely on the predictive variances, i.e., the diagonal elements of $\boldsymbol{\Sigma}^p$. Nevertheless, Cholesky-based computation of these variances still entails a substantial computational burden involving $\mathcal{O}(n^2 \cdot n_p)$ operations even when a Cholesky factor of $\widetilde{\boldsymbol{\Sigma}}$ has been precomputed.

2.1 Vecchia-inducing-points full-scale (VIF) approximation

The idea of the Vecchia-inducing-points full-scale (VIF) approximation is to decompose $b(\boldsymbol{s}) + \epsilon(\boldsymbol{s})$ into two parts $b(\boldsymbol{s}) + \epsilon(\boldsymbol{s}) = b_{\mathrm{l}}(\boldsymbol{s}) + b_{\mathrm{s}}(\boldsymbol{s})$, where $b_{\mathrm{l}}(\boldsymbol{s})$ is a low-rank predictive process modeling large-scale dependence and $b_{\mathrm{s}}(\boldsymbol{s}) = b(\boldsymbol{s}) + \epsilon(\boldsymbol{s}) - b_{\mathrm{l}}(\boldsymbol{s})$ is a residual process capturing the residual variation that is unexplained by $b_{\mathrm{l}}(\boldsymbol{s})$. Specifically, for a given set $\mathcal{S}^* = \{\boldsymbol{s}_1^*, ..., \boldsymbol{s}_m^*\}$ of m inducing points, or knots, the predictive process is given by

$$b_{\mathrm{l}}(\boldsymbol{s}) = \boldsymbol{c}\left(\boldsymbol{s}, \mathcal{S}^*\right)^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{b}^*, \tag{3}$$

where $\boldsymbol{b}^* = (b(\boldsymbol{s}_1^*), ..., b(\boldsymbol{s}_m^*))^{\mathrm{T}}$. Its covariance is

$$\mathrm{Cov}\left(b_{\mathrm{l}}(\boldsymbol{s}_i), b_{\mathrm{l}}(\boldsymbol{s}_j)\right) = \boldsymbol{c}(\boldsymbol{s}_i, \mathcal{S}^*)^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{c}\left(\boldsymbol{s}_j, \mathcal{S}^*\right),$$

where $\boldsymbol{c}(\boldsymbol{s}_i, \mathcal{S}^*) = \left(c_{\boldsymbol{\theta}}(\boldsymbol{s}_i, \boldsymbol{s}_1^*), ..., c_{\boldsymbol{\theta}}(\boldsymbol{s}_i, \boldsymbol{s}_m^*)\right)^{\mathrm{T}}$ and $\boldsymbol{\Sigma}_m = \left[c_{\boldsymbol{\theta}}(\boldsymbol{s}_i^*, \boldsymbol{s}_j^*)\right]_{i=1:m, j=1:m} \in \mathbb{R}^{m \times m}$ is a covariance matrix of the inducing points. The covariance matrix of $\boldsymbol{b}_{\mathrm{l}} = (b_{\mathrm{l}}(\boldsymbol{s}_1), \ldots, b_{\mathrm{l}}(\boldsymbol{s}_n))^{\mathrm{T}}$ is thus given by

$$\boldsymbol{\Sigma}^{\mathrm{l}} = \mathrm{Cov}(\boldsymbol{b}_{\mathrm{l}}) = \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn},$$

where $\boldsymbol{\Sigma}_{mn} = \left[c_{\boldsymbol{\theta}}(\boldsymbol{s}_i^*, \boldsymbol{s}_j)\right]_{i=1:m, j=1:n} \in \mathbb{R}^{m \times n}$ is a cross-covariance matrix between the inducing and data points. Concerning the residual process $b_{\mathrm{s}}(\boldsymbol{s})$, $\mathrm{Cov}(\boldsymbol{b}_{\mathrm{s}})$ is given by

$$\mathrm{Cov}(\boldsymbol{b}_{\mathrm{s}}) = \widetilde{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn},$$

where $\boldsymbol{b}_{\mathrm{s}} = (b_{\mathrm{s}}(\boldsymbol{s}_1), \ldots, b_{\mathrm{s}}(\boldsymbol{s}_n))^{\mathrm{T}}$. In a VIF approximation, this residual covariance matrix is approximated using a Vecchia approximation, which represents the full joint density $p(\boldsymbol{b}_{\mathrm{s}} \mid \boldsymbol{\theta})$ through a product of low-dimensional conditional densities,

$$p(\boldsymbol{b}_{\mathrm{s}} \mid \boldsymbol{\theta}) \approx \prod_{i=1}^{n} p(b_{\mathrm{s}}(\boldsymbol{s}_i) \mid \boldsymbol{b}_{\mathrm{s}_{N(i)}}, \boldsymbol{\theta}),$$

where $N(i) \subseteq \{1, \ldots, i-1\}$ are sets of conditioning points, often called neighbors in Vecchia approximations (Katzfuss and Guinness, 2021; Datta et al., 2016). This leads to a sparse approximate Cholesky factorization of the precision matrix $\mathrm{Cov}(\boldsymbol{b}_{\mathrm{s}})^{-1}$:

$$(\widetilde{\boldsymbol{\Sigma}}^{\mathrm{s}})^{-1} = \boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B} \approx \mathrm{Cov}(\boldsymbol{b}_{\mathrm{s}})^{-1},$$

where $\boldsymbol{D} = \text{diag}(D_i)$ and $\boldsymbol{B}$ is a sparse lower triangular matrix with 1's on the diagonal and non-zero off-diagonal entries $\boldsymbol{B}_{iN(i)} = -\boldsymbol{A}_i \in \mathbb{R}^{|N(i)|}$, and D_i and $\boldsymbol{A}_i$ are defined as

$$\begin{aligned} D_i &= \widetilde{\boldsymbol{\Sigma}}_{ii} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi} - \Big(\boldsymbol{A}_i\big(\boldsymbol{\Sigma}_{iN(i)}^{\mathrm{T}} - \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi}\big)\Big), \\ \boldsymbol{A}_i &= \big(\boldsymbol{\Sigma}_{iN(i)} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\big)\big(\widetilde{\boldsymbol{\Sigma}}_{N(i)} - \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\big)^{-1}, \end{aligned} \tag{4}$$

where $\boldsymbol{\Sigma}_{mN(i)} = \big[c_{\boldsymbol{\theta}}(\boldsymbol{s}_l^*, \boldsymbol{s}_k^{N_i})\big]_{l=1:m,k=1:|N(i)|} \in \mathbb{R}^{m\times|N(i)|}$, $\boldsymbol{s}_k^{N_i}$ are the Vecchia neighbor inputs of $\boldsymbol{s}_i$, and $\boldsymbol{\Sigma}_{mi} = \big[c_{\boldsymbol{\theta}}(\boldsymbol{s}_l^*, \boldsymbol{s}_i)\big]_{l=1:m} \in \mathbb{R}^m$. As is commonly done, we choose $N(i)$ as the indices of the m_v nearest (in a sense to be defined below) neighbors of $\boldsymbol{s}_i$ among $\boldsymbol{s}_1, \dots, \boldsymbol{s}_{i-1}$ if $i > m_v + 1$, and $N(i) = \{1, \dots, i-1\}$ if $i \leq m_v + 1$. Calculating a Vecchia approximation for the residual process has $\mathcal{O}\big(n \cdot (m_v^3 + m_v^2 \cdot m)\big)$ computational cost and requires $\mathcal{O}\big(n \cdot (m_v + m)\big)$ memory storage.

In summary, a VIF approximation for the covariance matrix $\widetilde{\boldsymbol{\Sigma}}$ in (2) is given by

$$\widetilde{\boldsymbol{\Sigma}}_\dagger = \boldsymbol{\Sigma}^{\mathrm{l}} + \widetilde{\boldsymbol{\Sigma}}^{\mathrm{s}} \approx \widetilde{\boldsymbol{\Sigma}}$$

and the corresponding negative log-likelihood is

$$\mathcal{L}_\dagger(\boldsymbol{\theta}; \boldsymbol{y}) = \frac{n}{2}\log(2\pi) + \frac{1}{2}\log\det\big(\widetilde{\boldsymbol{\Sigma}}_\dagger\big) + \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\widetilde{\boldsymbol{\Sigma}}_\dagger^{-1}\boldsymbol{y}.$$

Note that if the number of Vecchia neighbors is zero, the VIF approximation reduces to the FITC approximation as a special case, and if there are no inducing points, the VIF approximation coincides with a classical Vecchia approximation. In addition, the VIF approximation can be interpreted as a general Vecchia approximation (Katzfuss and Guinness, 2021), in which each conditional distribution conditions on all inducing points in addition to the local Vecchia neighbors. However, the inverse Cholesky factor of the VIF approximation is, in general, not sparse. Sparsity would only arise in the special case where the inducing points coincide with observation locations.

2.2 Computational complexity and calculation of gradients

For computational efficiency, we can apply the Sherman-Woodbury-Morrison formula:

$$\begin{aligned} \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1} &= (\widetilde{\boldsymbol{\Sigma}}^{\mathrm{s}} + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn})^{-1} \\ &= \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}(\boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}})^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}, \end{aligned}$$

where we denote

$$\boldsymbol{M} = \boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}. \tag{5}$$

Moreover, by Sylvester's determinant theorem, the determinant of $\widetilde{\boldsymbol{\Sigma}}_\dagger$ can be calculated as

$$\begin{aligned} \det\big(\widetilde{\boldsymbol{\Sigma}}_\dagger\big) &= \det\big(\widetilde{\boldsymbol{\Sigma}}^{\mathrm{s}} + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\big) \\ &= \det\big(\boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\big) \cdot \det\big(\boldsymbol{\Sigma}_m\big)^{-1} \cdot \det\big(\boldsymbol{D}\big). \end{aligned}$$

If a first- or second-order method for convex optimization, such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Fletcher, 2000) is used, gradients of the negative log-likelihood function are required. For the VIF approximation, the gradient with respect to $\boldsymbol{\theta}$ is given by

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_\dagger(\boldsymbol{\theta}; \boldsymbol{y}) = \frac{1}{2} \operatorname{Tr}\left(\widetilde{\boldsymbol{\Sigma}}_\dagger^{-1} \frac{\partial \widetilde{\boldsymbol{\Sigma}}_\dagger}{\partial \boldsymbol{\theta}}\right) - \frac{1}{2} \boldsymbol{y}^{\mathrm{T}} \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1} \frac{\partial \widetilde{\boldsymbol{\Sigma}}_\dagger}{\partial \boldsymbol{\theta}} \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1} \boldsymbol{y},$$

where $\frac{\partial \widetilde{\boldsymbol{\Sigma}}_\dagger}{\partial \boldsymbol{\theta}} = \frac{\partial \widetilde{\boldsymbol{\Sigma}}^{\mathrm{s}}}{\partial \boldsymbol{\theta}} + \frac{\partial \boldsymbol{\Sigma}^{\mathrm{l}}}{\partial \boldsymbol{\theta}}$ with

$$\begin{aligned}
\frac{\partial \widetilde{\boldsymbol{\Sigma}}^{\mathrm{s}}}{\partial \boldsymbol{\theta}} &= \boldsymbol{B}^{-1} \frac{\partial \boldsymbol{D}}{\partial \boldsymbol{\theta}} \boldsymbol{B}^{-\mathrm{T}} - \boldsymbol{B}^{-1} \frac{\partial \boldsymbol{B}}{\partial \boldsymbol{\theta}} \boldsymbol{B}^{-1} \boldsymbol{D} \boldsymbol{B}^{-\mathrm{T}} - \boldsymbol{B}^{-1} \boldsymbol{D} \boldsymbol{B}^{-\mathrm{T}} \frac{\partial \boldsymbol{B}^{\mathrm{T}}}{\partial \boldsymbol{\theta}} \boldsymbol{B}^{-\mathrm{T}}, \\
\frac{\partial \boldsymbol{\Sigma}^{\mathrm{l}}}{\partial \boldsymbol{\theta}} &= \frac{\partial \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}}{\partial \boldsymbol{\theta}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \frac{\partial \boldsymbol{\Sigma}_{mn}}{\partial \boldsymbol{\theta}} - \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \frac{\partial \boldsymbol{\Sigma}_m}{\partial \boldsymbol{\theta}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn}.
\end{aligned}$$

In Appendix A, we derive the gradients of $\boldsymbol{B}$ and $\boldsymbol{D}$.

The computational complexity associated with the computation of the negative log-likelihood and its derivatives is of the order $\mathcal{O}\big(n \cdot (m_v^3 + m_v^2 \cdot m + m^2)\big)$ and the required storage of order $\mathcal{O}\big(n \cdot (m + m_v)\big)$.

2.3 Prediction with VIF approximations

We propose to do predictions for the response $\boldsymbol{y}^p$ at n_p prediction inputs $\mathcal{S}^p = \{\boldsymbol{s}_1^p, \dots, \boldsymbol{s}_{n_p}^p\}$ with VIF approximations using the following result.

Proposition 1 *A VIF-approximated posterior predictive distribution for $p(\boldsymbol{y}^p|\boldsymbol{y}, \boldsymbol{\theta})$, $\boldsymbol{y}^p = (y(\boldsymbol{s}_1^p), \dots, y(\boldsymbol{s}_{n_p}^p))^{\mathrm{T}} \in \mathbb{R}^{n_p}$, is given by $\mathcal{N}\big(\boldsymbol{\mu}_\dagger^p, \boldsymbol{\Sigma}_\dagger^p\big)$, where*

$$\begin{aligned}
\boldsymbol{\mu}_\dagger^p &= -\boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} \boldsymbol{y} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} \boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B} \boldsymbol{y} \\
&\quad - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} \boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B} \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{M}^{-1} \boldsymbol{\Sigma}_{mn} \boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B} \boldsymbol{y} \qquad (6) \\
&\quad + \boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{M}^{-1} \boldsymbol{\Sigma}_{mn} \boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B} \boldsymbol{y} \\
\boldsymbol{\Sigma}_\dagger^p &= \boldsymbol{B}_p^{-1} \boldsymbol{D}_p \boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1} \boldsymbol{B}_{po}^{\mathrm{T}} \boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} \qquad (7) \\
&\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1}\big) \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1} \\
&\quad \cdot \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1} \boldsymbol{B}_{po}^{\mathrm{T}} \boldsymbol{B}_p^{-\mathrm{T}}\big),
\end{aligned}$$

where $\boldsymbol{\Sigma}_{mn_p} = \big[c_{\boldsymbol{\theta}}(\boldsymbol{s}_i^*, \boldsymbol{s}_j^p)\big]_{i=1:m, j=1:n_p} \in \mathbb{R}^{m \times n_p}$, and $\boldsymbol{B}_{po} \in \mathbb{R}_p^{n_p \times n}$, $\boldsymbol{B}_p \in \mathbb{R}_p^{n_p \times n_p}$, and $\boldsymbol{D}_p^{-1} \in \mathbb{R}^{n_p \times n_p}$ are defined below in (8).

Proof [Proof of Proposition 1] First, note that $(\boldsymbol{y}^{\mathrm{T}}, \boldsymbol{y}^{p\mathrm{T}})^{\mathrm{T}} = (\boldsymbol{b}_{\mathrm{l}}^{\mathrm{T}}, \boldsymbol{b}_{\mathrm{l}}^{p\mathrm{T}})^{\mathrm{T}} + (\boldsymbol{b}_{\mathrm{s}}^{\mathrm{T}}, \boldsymbol{b}_{\mathrm{s}}^{p\mathrm{T}})^{\mathrm{T}}$, where $\boldsymbol{b}_{\mathrm{l}}^p$ and $\boldsymbol{b}_{\mathrm{s}}^p$ denote the low-rank and residual processes, respectively, at the prediction points $\mathcal{S}^p$. Following common practice for predictions with Vecchia approximations (Katzfuss et al., 2020), we apply a Vecchia approximation to the joint distribution of the residual process $(\boldsymbol{b}_{\mathrm{s}}^{\mathrm{T}}, \boldsymbol{b}_{\mathrm{s}}^{p\mathrm{T}})^{\mathrm{T}}$ at the training and prediction inputs whose covariance matrix is

$$\operatorname{Cov}\left((\boldsymbol{b}_{\mathrm{s}}^{\mathrm{T}}, \boldsymbol{b}_{\mathrm{s}}^{p\mathrm{T}})^{\mathrm{T}}\right) = \begin{pmatrix} \widetilde{\boldsymbol{\Sigma}} & \boldsymbol{\Sigma}_{nn_p} \\ \boldsymbol{\Sigma}_{nn_p}^{\mathrm{T}} & \widetilde{\boldsymbol{\Sigma}}_p \end{pmatrix} - \begin{pmatrix} \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} & \boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} \\ \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} & \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} \end{pmatrix}.$$

This gives

$$\mathrm{Cov}\left((\boldsymbol{b}_{\mathrm{s}}^{\mathrm{T}}, \boldsymbol{b}_{\mathrm{s}}^{p\mathrm{T}})^{\mathrm{T}}\right)^{-1} \approx \begin{pmatrix} \boldsymbol{B} & \boldsymbol{0} \\ \boldsymbol{B}_{po} & \boldsymbol{B}_p \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \boldsymbol{D}^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_p^{-1} \end{pmatrix} \begin{pmatrix} \boldsymbol{B} & \boldsymbol{0} \\ \boldsymbol{B}_{po} & \boldsymbol{B}_p \end{pmatrix}. \tag{8}$$

It follows that

$$\begin{aligned} &\mathrm{Cov}\left((\boldsymbol{b}_{\mathrm{s}}^{\mathrm{T}}, \boldsymbol{b}_{\mathrm{s}}^{p\mathrm{T}})^{\mathrm{T}}\right) \approx \\ &\begin{pmatrix} (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} & -(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\ -\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} & (\boldsymbol{B}_p^{\mathrm{T}}\boldsymbol{D}_p^{-1}\boldsymbol{B}_p)^{-1} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \end{pmatrix} \end{aligned}$$

and thus

$$\begin{aligned} \mathrm{Cov}\left((\boldsymbol{y}^{\mathrm{T}}, \boldsymbol{y}^{p\mathrm{T}})^{\mathrm{T}}\right) &\approx \begin{pmatrix} \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} & \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\ \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} & \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \end{pmatrix} \\ &+ \begin{pmatrix} (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} & -(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\ -\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} & (\boldsymbol{B}_p^{\mathrm{T}}\boldsymbol{D}_p^{-1}\boldsymbol{B}_p)^{-1} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \end{pmatrix}. \end{aligned}$$

By standard arguments for conditional probabilities of multivariate Gaussian distributions, we obtain $\boldsymbol{y}^p \mid \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}_\dagger^p, \boldsymbol{\Sigma}_\dagger^p\right)$ in Proposition 1. ■

In Appendix C, we additionally derive an alternative and equivalent expression for $\boldsymbol{\Sigma}_\dagger^p$, which allows for a more efficient calculation and is used in our software implementation. The predictive distribution $\boldsymbol{b}^p|\boldsymbol{y}$ of the latent GP $\boldsymbol{b}^p$ is obtained analogously by subtracting $\sigma^2\boldsymbol{I}_{n_p}$ from $\boldsymbol{\Sigma}_\dagger^p$ in (7). The computational costs for the predictive means $\boldsymbol{\mu}_\dagger^p$ and variances $\mathrm{diag}(\boldsymbol{\Sigma}_\dagger^p)$ are $\mathcal{O}\big(n_p\cdot(m_v^3+m_v^2\cdot m)+n\cdot(m_v+m)\big)$ and $\mathcal{O}\big(n_p\cdot(m_v+m\cdot m_v)+n\cdot(m\cdot m_v+m^2)\big)$, respectively.

3. VIF-Laplace approximations for latent Gaussian process models

In the following, we assume that the response variable $\boldsymbol{y} = (y(\boldsymbol{s}_1), \dots, y(\boldsymbol{s}_n))^{\mathrm{T}} \in \mathbb{R}^n$ follows a parametric distribution with a density of $p(\boldsymbol{y} \mid \boldsymbol{\mu}, \boldsymbol{\xi}) = \prod_{i=1}^n p(y_i|\mu_i, \xi)$ with respect to a sigma finite product measure conditional on $\boldsymbol{\mu} = F(\boldsymbol{X})+\boldsymbol{b}$. This density has, potentially link function-transformed, parameters $\boldsymbol{\mu} \in \mathbb{R}^n$ and additional auxiliary parameters $\boldsymbol{\xi} \in \Xi \subset \mathbb{R}^r$. For instance, μ_i and ξ can be the mean and variance of a Gaussian distribution, the log-mean and the shape parameter of a gamma likelihood, or μ_i can be the logit-transformed success probability of a Bernoulli distribution for which there is no additional parameter ξ. Similarly as in (1), the parameter $\boldsymbol{\mu}$ is modeled as the sum of fixed effects $F(\boldsymbol{X})$ and a finite-dimensional version of a GP $\boldsymbol{b}$, $\boldsymbol{\mu} = F(\boldsymbol{X})+\boldsymbol{b}$, where $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ contains predictor variables and $\boldsymbol{b} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$. We again assume for simplicity that $F(\boldsymbol{X}) = \boldsymbol{0}$, but this assumption can be easily relaxed. Estimation of the covariance $\boldsymbol{\theta} \in \mathbb{R}^q$ and auxiliary parameters $\boldsymbol{\xi}$ is done by minimizing the negative log-marginal likelihood

$$-\log\big(p(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\xi})\big) = -\log\Big(\int p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi})p(\boldsymbol{b} \mid \boldsymbol{\theta})d\boldsymbol{b}\Big). \tag{9}$$

For non-Gaussian likelihoods, there is typically no analytic expression for $p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi})$ and an approximation has to be used. In this article, we use the Laplace approximation (Tierney and Kadane, 1986), additionally assuming that $p(\boldsymbol{y} \mid \boldsymbol{\mu}, \boldsymbol{\xi})$ is log-concave in $\boldsymbol{\mu}$. The Laplace approximation for (9) is given by

$$L^{LA}(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}) = -\log p\left(\boldsymbol{y} \mid \tilde{\boldsymbol{b}}, \boldsymbol{\xi}\right) + \frac{1}{2}\tilde{\boldsymbol{b}}^T \boldsymbol{\Sigma}^{-1}\tilde{\boldsymbol{b}} + \frac{1}{2}\log\det\left(\boldsymbol{\Sigma}\boldsymbol{W} + \boldsymbol{I}_n\right), \tag{10}$$

where $\tilde{\boldsymbol{b}} = \operatorname{argmax}_{\boldsymbol{b}} \log p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi}) - \frac{1}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{b}$ is the mode of $p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi})p(\boldsymbol{b} \mid \boldsymbol{\theta})$, and $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ is diagonal with

$$W_{ii} = -\left.\frac{\partial^2 \log p\left(y(\boldsymbol{s}_i) \mid b(\boldsymbol{s}_i), \boldsymbol{\xi}\right)}{\partial b(\boldsymbol{s}_i)^2}\right|_{\boldsymbol{b}=\tilde{\boldsymbol{b}}}. \tag{11}$$

The mode $\tilde{\boldsymbol{b}}$ is usually found with Newton's method (Rasmussen and Williams, 2006). Gradients of $L^{LA}(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi})$ for parameter estimation can be found, e.g., in Rasmussen and Williams (2006) and Sigrist (2022b). We apply a VIF approximation to the latent process $\boldsymbol{b}$ by again decomposing it into a low-rank and a residual process, $\boldsymbol{b} = \boldsymbol{b}_{\mathrm{l}} + \boldsymbol{b}_{\mathrm{s}}$, where $\boldsymbol{b}_{\mathrm{l}}$ and $\boldsymbol{b}_{\mathrm{s}}$ are defined as in Section 2.1 except that the residual process $\boldsymbol{b}_{\mathrm{s}}$ does not contain any error variance. Specifically, the precision matrix $\mathrm{Cov}(\boldsymbol{b}_{\mathrm{s}})^{-1} = (\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn})^{-1}$ is approximated as

$$(\boldsymbol{\Sigma}^{\mathrm{s}})^{-1} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} \approx \mathrm{Cov}(\boldsymbol{b}_{\mathrm{s}})^{-1},$$

where $\boldsymbol{B}$ and $\boldsymbol{D}$ are defined analogously as in Section 2.1, but $\widetilde{\boldsymbol{\Sigma}}_{ii}$ and $\widetilde{\boldsymbol{\Sigma}}_{N(i)}$ are replaced by $\boldsymbol{\Sigma}_{ii}$ and $\boldsymbol{\Sigma}_{N(i)}$ not including the error variance in (4). A VIF-Laplace approximation (VIFLA) is then given by combining the VIF and Laplace approximations to obtain the following approximate negative log-marginal likelihood $-\log\left(p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi})\right)$:

$$L^{VIFLA}(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}) = -\log p\left(\boldsymbol{y} \mid \tilde{\boldsymbol{b}}, \boldsymbol{\xi}\right) + \frac{1}{2}\tilde{\boldsymbol{b}}^T \boldsymbol{\Sigma}_\dagger^{-1}\tilde{\boldsymbol{b}} + \frac{1}{2}\log\det\left(\boldsymbol{\Sigma}_\dagger\boldsymbol{W} + \boldsymbol{I}_n\right), \tag{12}$$

where $\tilde{\boldsymbol{b}} = \operatorname{argmax}_{\boldsymbol{b}} \log p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi}) - \frac{1}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{\Sigma}_\dagger^{-1}\boldsymbol{b}$ is the mode of $p(\boldsymbol{y} \mid \boldsymbol{b}, \boldsymbol{\xi})p_\dagger(\boldsymbol{b} \mid \boldsymbol{\theta})$ with $p_\dagger(\boldsymbol{b} \mid \boldsymbol{\theta})$ being the density of $\mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\dagger)$ and

$$\boldsymbol{\Sigma}_\dagger^{-1} = (\boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T} + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn})^{-1} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B},$$

where we have applied the Sherman-Woodbury-Morrison formula, and we recall that $\boldsymbol{M}$ is defined in (5). By properties of the determinant of matrix products and Sylvester's determinant theorem, we have

$$\begin{aligned}\log\det(\boldsymbol{\Sigma}_\dagger\boldsymbol{W} + \boldsymbol{I}_n) = &-\log\det(\boldsymbol{\Sigma}_m) - \log\det(\boldsymbol{D}^{-1}) + \log\det(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}) \\ &+ \log\det\left(\boldsymbol{M} - \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\right),\end{aligned}$$

see Appendix B for the detailed derivation. Furthermore, one iteration of Newton's method is given by

$$\tilde{\boldsymbol{b}}^{t+1} = \left(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1}\right)^{-1}\left(\boldsymbol{W}\tilde{\boldsymbol{b}}^t + \frac{\partial \log p\left(\boldsymbol{y} \mid \tilde{\boldsymbol{b}}^t, \boldsymbol{\xi}\right)}{\partial \boldsymbol{b}}\right), t = 0, 1, \ldots \tag{13}$$

Applying the Sherman-Woodbury-Morrison formula, we can do linear solves with $\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1}$ using the following relationship:

$$\begin{aligned}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1} = \boldsymbol{W}^{-1}\big(&\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W} - \boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\\ &\cdot \boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}(\boldsymbol{\Sigma}_m+\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}})^{-1}\\ &\cdot \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\big)\boldsymbol{\Sigma}_\dagger,\end{aligned} \tag{14}$$

see Appendix B for a detailed derivation. The gradients of $L^{VIFLA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})$, with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\xi}$ are also derived in Appendix B. As the above results show, the computational complexity of evaluating the negative log-marginal likelihood and its derivatives is primarily determined by the Cholesky factorization of the matrix $\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}$. Despite that $\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}$ is sparse, this is computationally expensive for large n as mentioned in the introduction. For two-dimensional spatial data, the Cholesky decomposition for $\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}$ has complexity of approximately $\mathcal{O}(n^{3/2})$, while in higher dimensions, it can be $\mathcal{O}(n^2)$ or larger (Lipton et al., 1979).

3.1 Prediction with VIF-Laplace approximations

The goal is to predict either the latent GP $\boldsymbol{b}^p \in \mathbb{R}^{n_p}$ or the response variable $\boldsymbol{y}^p \in \mathbb{R}^{n_p}$ at n_p new inputs $\mathcal{S}^p = \{\boldsymbol{s}_1^p,\dots,\boldsymbol{s}_{n_p}^p\}$ using the posterior predictive distributions $p\left(\boldsymbol{b}^p \mid \boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi}\right)$ and $p\left(\boldsymbol{y}^p \mid \boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi}\right)$, respectively. For the former, we have the following result.

Proposition 2 *A VIF-Laplace-approximated posterior predictive distribution for* $p\left(\boldsymbol{b}^p \mid \boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi}\right)$, $\boldsymbol{b}^p = (b(\boldsymbol{s}_1^p),\dots,b(\boldsymbol{s}_{n_p}^p))^{\mathrm{T}} \in \mathbb{R}^{n_p}$, *is given by* $\mathcal{N}\left(\boldsymbol{\omega}_p,\boldsymbol{\Omega}_p\right)$, *where*

$$\begin{aligned}\boldsymbol{\omega}_p &= -\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\tilde{\boldsymbol{b}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\tilde{\boldsymbol{b}}\\ &\quad - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\tilde{\boldsymbol{b}}\\ &\quad + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\tilde{\boldsymbol{b}}\\ \boldsymbol{\Omega}_p &= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\ &\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\big(\boldsymbol{\Sigma}_\dagger^{-1} - \boldsymbol{\Sigma}_\dagger^{-1}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\boldsymbol{\Sigma}_\dagger^{-1}\big)\\ &\quad \cdot \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big),\end{aligned} \tag{15}$$

and $\boldsymbol{B}_{po} \in \mathbb{R}_p^{n_p\times n}$, $\boldsymbol{B}_p \in \mathbb{R}_p^{n_p\times n_p}$, *and* $\boldsymbol{D}_p^{-1} \in \mathbb{R}^{n_p\times n_p}$ *are defined analogously as in* (8).

Proof [Proof of Proposition 2]

This can be derived as follows. As in Section 2.3, a Vecchia approximation is applied to the joint distribution of the residual GP at the training and prediction inputs, and we obtain $\boldsymbol{b}^p \mid \boldsymbol{b} \sim \mathcal{N}\left(\boldsymbol{v}_p,\boldsymbol{\Xi}_p\right)$ with

$$\begin{aligned}\boldsymbol{v}_p &= -\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{b} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{b}\\ &\quad - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{b}\\ &\quad + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{b}\\ \boldsymbol{\Xi}_p &= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}.\end{aligned}$$

By the law of total probability, we have

$$p\left(\boldsymbol{b}^p \mid \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}\right) = \int p\left(\boldsymbol{b}^p \mid \boldsymbol{b}, \boldsymbol{\theta}\right) p(\boldsymbol{b} \mid \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}) d\boldsymbol{b}.$$

The Laplace approximation $p(\boldsymbol{b} \mid \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}) \approx \mathcal{N}\big(\tilde{\boldsymbol{b}}, (\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)$ and $\boldsymbol{b}^p \mid \boldsymbol{b} \sim \mathcal{N}\left(\boldsymbol{v}_p, \boldsymbol{\Xi}_p\right)$ then give the result in (15). ■

See Appendix C for an alternative and equivalent expression for $\boldsymbol{\Omega}_p$, which allows for a more efficient calculation and is used in our software implementation. For predicting the response variable, the integral $p\left(\boldsymbol{y}^p \mid \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}\right) = \int p\left(\boldsymbol{y}^p \mid \boldsymbol{b}^p, \boldsymbol{\xi}\right) p\left(\boldsymbol{b}^p \mid \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}\right) d\boldsymbol{b}^p$ is analytically intractable for most likelihoods and must be approximated using numerical integration or by simulating from $p\left(\boldsymbol{b}^p \mid \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\xi}\right) \approx \mathcal{N}\left(\boldsymbol{\omega}_p, \boldsymbol{\Omega}_p\right)$.

4. Iterative methods for VIF-Laplace approximations

While VIF-Laplace approximations improve scalability, a sparse Cholesky factorization of the matrix $\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}$ remains computationally expensive as mentioned above. To address this, we next show how iterative methods and stochastic approximations can be used for fast computations. Parameter estimation and prediction with VIF-Laplace approximations involve several computationally expensive operations outlined in the following. First, calculating linear solves $(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})\boldsymbol{u} = \boldsymbol{v}$, $\boldsymbol{v} \in \mathbb{R}^n$, (i) in Newton's method for finding the mode, see (13), (ii) for implicit derivatives of the log-marginal likelihood such as $(\frac{\partial L^{VIFLA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})}{\partial \tilde{\boldsymbol{b}}})^{\mathrm{T}} \frac{\partial \tilde{\boldsymbol{b}}}{\partial \theta_k}$, see Appendix B, and (iii) for predictive distributions in (15). The latter is particularly challenging as the number of prediction points n_p is typically large, and the linear systems thus contain many right-hand sides, see Section 4.2 for more information. Second, calculating log-determinants $\log\det\left(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \mathbf{I}_n\right)$ in the log-marginal likelihood given in (12). And third, calculating trace terms such as $\mathrm{Tr}((\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})^{-1} \frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial \theta_k}) = \frac{\partial \log\det\left(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1}\right)}{\partial \theta_k}$ for the derivatives of log-determinants, see Appendix B.

4.1 Likelihood evaluation and gradient calculation

To compute linear solves $(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})\boldsymbol{u} = \boldsymbol{v}$, we apply the preconditioned conjugate gradient (CG) method (Saad, 2003). Preconditioning increases the convergence speed of the CG method, and it also reduces the variance of stochastic log-determinant and gradient approximations. Below in Section 4.3, we propose two preconditioners. Preconditioning means that we solve the equivalent system

$$\boldsymbol{P}^{-\frac{1}{2}}(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})\boldsymbol{P}^{-\frac{T}{2}}\boldsymbol{P}^{\frac{T}{2}}\boldsymbol{u} = \boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{v}, \tag{16}$$

where $\boldsymbol{P}$ is a symmetric positive definite preconditioner matrix. Alternatively, since $\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1} = \boldsymbol{\Sigma}_\dagger^{-1}(\boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_\dagger)\boldsymbol{W}$, we can equivalently solve

$$\boldsymbol{P}^{-\frac{1}{2}}(\boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_\dagger)\boldsymbol{P}^{-\frac{T}{2}}\boldsymbol{P}^{\frac{T}{2}}\boldsymbol{W}\boldsymbol{u} = \boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{\Sigma}_\dagger\boldsymbol{v}. \tag{17}$$

The version in (16) is used for the VIFDU preconditioner defined in Section 4.3.1, while (17) is used for the FITC preconditioner defined in Section 4.3.2. In each iteration of the

preconditioned CG method for (16) and (17), we calculate matrix-vector products of the form $(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})\boldsymbol{w}$ and $(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_{m}^{-1}\boldsymbol{\Sigma}_{mn})\boldsymbol{w}$, respectively, $\boldsymbol{w} \in \mathbb{R}^n$, which require $\mathcal{O}\big(n \cdot (m + m_v)\big)$ operations. The computational complexity for computing and inverting the matrix $\boldsymbol{M}$ is $\mathcal{O}\big(n \cdot (m^2 + m \cdot m_v)\big)$.

To calculate $\log\det\big(\boldsymbol{\Sigma}_\dagger\boldsymbol{W} + \mathbf{I}_n\big)$, we employ the stochastic Lanczos quadrature (SLQ) method (Ubaru et al., 2017) which combines a quadrature technique based on the Lanczos algorithm (Lanczos, 1950) with Hutchinson's stochastic trace estimator (Hutchinson, 1989). Dong et al. (2017) analyzed different approaches to estimate log-determinants and found that the SLQ method is superior to other methods. The preconditioned SLQ method gives the following approximation:

$$\log\det\big(\boldsymbol{\Sigma}_\dagger\boldsymbol{W} + \mathbf{I}_n\big) \approx \log\det\big(\boldsymbol{\Sigma}_\dagger\big) + \frac{n}{\ell}\sum_{i=1}^{\ell} \boldsymbol{e}_1^{\mathrm{T}} \log\big(\widetilde{\boldsymbol{T}}_i\big)\boldsymbol{e}_1 + \log\det(\boldsymbol{P}), \tag{18}$$

where $\widetilde{\boldsymbol{T}}_i \in \mathbb{R}^{k\times k}$ is the tridiagonal matrix of the Lanczos tridiagonalization $\widetilde{\boldsymbol{Q}}_i\widetilde{\boldsymbol{T}}_i\widetilde{\boldsymbol{Q}}_i^{\mathrm{T}} \approx \boldsymbol{P}^{-\frac{1}{2}}(\boldsymbol{W} + \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1})\boldsymbol{P}^{-\frac{1}{2}}$ obtained by running the Lanczos algorithm for k steps with the matrix $\boldsymbol{P}^{-\frac{1}{2}}(\boldsymbol{W} + \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1})\boldsymbol{P}^{-\frac{1}{2}}$ and initial vector $\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{z}_i/\sqrt{n}$, where $\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{P})$ and $\sqrt{n}$ approximates the normalization factor $\|\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{z}_i\|_2$. See Appendix D for a detailed derivation. Alternatively, we can obtain the following SLQ approximation:

$$\log\det\big(\boldsymbol{\Sigma}_\dagger\boldsymbol{W} + \mathbf{I}_n\big) \approx \log\det\big(\boldsymbol{W}\big) + \frac{n}{\ell}\sum_{i=1}^{\ell} \boldsymbol{e}_1^{\mathrm{T}} \log\big(\widetilde{\boldsymbol{T}}_i\big)\boldsymbol{e}_1 + \log\det(\boldsymbol{P}), \tag{19}$$

where now $\widetilde{\boldsymbol{T}}_i \in \mathbb{R}^{k\times k}$ is the tridiagonal matrix of the partial Lanczos tridiagonalization $\widetilde{\boldsymbol{Q}}_i\widetilde{\boldsymbol{T}}_i\widetilde{\boldsymbol{Q}}_i^{\mathrm{T}} \approx \boldsymbol{P}^{-\frac{1}{2}}(\boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_\dagger)\boldsymbol{P}^{-\frac{1}{2}}$ obtained by running the Lanczos algorithm for k steps with the matrix $\boldsymbol{P}^{-\frac{1}{2}}(\boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_\dagger)\boldsymbol{P}^{-\frac{1}{2}}$ and initial vector $\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{z}_i/\|\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{z}_i\|_2$, $\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{P})$. The version in (18) is used for the VIFDU preconditioner defined in Section 4.3.1, whereas (19) is used for the FITC preconditioner defined in Section 4.3.2.

As in Gardner et al. (2018a), we use a technique from Saad (2003) to calculate the partial Lanczos tridiagonal matrices $\widetilde{\boldsymbol{T}}_1, \dots, \widetilde{\boldsymbol{T}}_l$ from the coefficients of the preconditioned CG algorithm when solving $(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})^{-1}\boldsymbol{z}_i$, or $(\boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_\dagger)^{-1}\boldsymbol{z}_i$, $i = 1, \dots, l$. In doing so, we avoid running the Lanczos algorithm, which brings multiple advantages: Numerical instabilities are not an issue and storing $\widetilde{\boldsymbol{Q}}_i$ is not necessary. In addition to the ℓ partial Lanczos tridiagonal matrices, the modified CG method computes the ℓ linear solves $(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})^{-1}\boldsymbol{z}_i$, or $(\boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_\dagger)^{-1}\boldsymbol{z}_i$, respectively, for the probe vectors $\boldsymbol{z}_i$. This brings the advantage that once the log-determinant is calculated, its gradients can be calculated with minimal computational overhead. Specifically, the trace terms for calculating derivatives of log-determinants given in Appendix B can be computed using stochastic trace estimation

(STE) as follows:

$$\begin{aligned}
\operatorname{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial b^*(\boldsymbol{s}_i)}\Big) &= \operatorname{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial b^*(\boldsymbol{s}_i)}\mathbb{E}_{\boldsymbol{z}_i\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{P})}\left[\boldsymbol{P}^{-1}\boldsymbol{z}_i\boldsymbol{z}_i^{\mathrm{T}}\right]\Big) \\
&\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial\boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big) \quad i=1,\dots,p, \\
\operatorname{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\theta_k}\Big) &\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial\boldsymbol{\Sigma}_\dagger^{-1}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big) \quad k=1,\dots,q, \\
\operatorname{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\xi_l}\Big) &\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial\boldsymbol{W}}{\partial\xi_l}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big) \quad l=1,\dots,r,
\end{aligned}$$

see Appendix D for a detailed derivation. We additionally apply variance reduction by using the preconditioner $\boldsymbol{P}$ to construct a control variate (Lemieux, 2014), see Appendix D for more information. We choose the Lanczos rank k adaptively by running the preconditioned CG algorithm until it has converged.

In each iteration of the preconditioned CG method, we calculate matrix-vector products, which involve $\mathcal{O}\big(n\cdot(m+m_v)\big)$ operations. In total, we have computational cost for calculating the negative log-marginal likelihood and its derivatives of $\mathcal{O}\big(n\cdot(m_v^3+m^2+m\cdot t+m_v\cdot t+m\cdot m_v^2)\big)$, where t is the number of iterations of the CG algorithm.

4.2 Predictive variances

Calculating predictive (co-)variances given in (15) is computationally expensive when n and n_p are large, even with iterative methods due to n_p right-hand sides. In the following, we propose two computationally efficient simulation-based approaches both relying on iterative methods to compute $\operatorname{diag}(\boldsymbol{\Omega}_p)$. For this, we split $\boldsymbol{\Omega}_p$ given in (15) into two parts:

$$\begin{aligned}
&\boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T}+\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}+\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad-\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}-\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\boldsymbol{\Sigma}_\dagger^{-1} \\
&\quad\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}-(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big)
\end{aligned} \tag{20}$$

and

$$\begin{aligned}
&\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}-\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\boldsymbol{\Sigma}_\dagger^{-1}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\boldsymbol{\Sigma}_\dagger^{-1} \\
&\quad\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}-(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big),
\end{aligned} \tag{21}$$

where the diagonal of the first part in (20) can be computed efficiently in a deterministic manner without using iterative methods, because the term $\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}$ cancels out.

In Algorithm 1, the second term in (21) is approximated stochastically by sampling from a normal distribution with the matrix in (21) as its covariance matrix. This algorithm provides an unbiased and consistent approximation for $\boldsymbol{\Omega}_p$; see Appendix C for the proof of Proposition 3. It can also be adapted to compute only the predictive variances $\operatorname{diag}(\boldsymbol{\Omega}_p)$ by summing $\boldsymbol{z}_i^{(8)}\circ\boldsymbol{z}_i^{(8)}$ in Line 9 and extracting the diagonal of each term in Line 11. To

compute the linear solve $(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\boldsymbol{z}_i^{(6)}$, or $\boldsymbol{W}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}\boldsymbol{\Sigma}_\dagger\boldsymbol{z}_i^{(6)}$, in Line 7, we use the preconditioned CG method.

Algorithm 1 Simulation-based Predictive (Co-)Variance Estimator (SBPV)

Require: Matrices $\boldsymbol{B}_{po}$, $\boldsymbol{D}_p$, $\boldsymbol{D}^{-1}$, $\boldsymbol{B}$, $\boldsymbol{B}_p^{-1}$, $\boldsymbol{\Sigma}_{mn}$, $\boldsymbol{\Sigma}_m$, $\boldsymbol{\Sigma}_{mn_p}$, $\boldsymbol{M}$, $\boldsymbol{W}$
Ensure: Approximated predictive covariances $\boldsymbol{\Omega}_p$

1: Initialize: $\boldsymbol{Z} \leftarrow \boldsymbol{0} \in \mathbb{R}^{n_p \times n_p}$
2: **for** $i = 1$ to ℓ **do**
3: Let $\boldsymbol{z}_i^{(1)}$, $\boldsymbol{z}_i^{(2)} \sim \mathcal{N}(0, \mathbf{I}_n)$ and $\boldsymbol{z}_i^{(3)} \sim \mathcal{N}(0, \mathbf{I}_m)$
4: $\boldsymbol{z}_i^{(4)} \leftarrow \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-\frac{1}{2}}\boldsymbol{z}_i^{(3)} + \boldsymbol{B}^{-1}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{z}_i^{(1)}$ $\triangleright$ $\boldsymbol{z}_i^{(4)} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\dagger)$
5: $\boldsymbol{z}_i^{(5)} \leftarrow \big(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\big)\boldsymbol{z}_i^{(4)}$ $\triangleright$ $\boldsymbol{z}_i^{(5)} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\dagger^{-1})$
6: $\boldsymbol{z}_i^{(6)} \leftarrow \boldsymbol{z}_i^{(5)} + \boldsymbol{W}^{\frac{1}{2}}\boldsymbol{z}_i^{(2)}$ $\triangleright$ $\boldsymbol{z}_i^{(6)} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})$
7: $\boldsymbol{z}_i^{(7)} \leftarrow \begin{cases} (\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\boldsymbol{z}_i^{(6)} \\ \text{or } \boldsymbol{W}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}\boldsymbol{\Sigma}_\dagger\boldsymbol{z}_i^{(6)} \end{cases}$ $\triangleright$ $\boldsymbol{z}_i^{(7)} \sim \mathcal{N}\big(0, (\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\big)$
8: $\boldsymbol{z}_i^{(8)} \leftarrow \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\boldsymbol{\Sigma}_\dagger^{-1}\boldsymbol{z}_i^{(7)}$
$\triangleright$ $\boldsymbol{z}_i^{(8)} \sim \mathcal{N}\Big(0, \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\boldsymbol{\Sigma}_\dagger^{-1}\big(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W}\big)^{-1}\boldsymbol{\Sigma}_\dagger^{-1}$
$\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big)\Big)$
9: $\boldsymbol{Z} \leftarrow \boldsymbol{Z} + \boldsymbol{z}_i^{(8)}(\boldsymbol{z}_i^{(8)})^{\mathrm{T}}$
10: **end for**
11: $\boldsymbol{\Omega}_p \leftarrow \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}$
$+ \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}$
$+ (\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p})^{\mathrm{T}} - (\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}})^{\mathrm{T}}$
$+ \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}$
$+ \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} + \frac{1}{\ell}\boldsymbol{Z}$

Proposition 3 *Algorithm 1 produces an unbiased and consistent estimator for the predictive covariance matrix* $\boldsymbol{\Omega}_p$.

In Algorithm 2, we estimate the diagonal of matrix in (21) using the stochastic approach proposed by Bekas et al. (2007), which approximates the diagonal of a matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ as $\mathrm{diag}(\mathbf{A}) \approx \sum_{i=1}^{\ell} \boldsymbol{z}_i \circ \mathbf{A}\boldsymbol{z}_i$, where $\boldsymbol{z}_i$ are Rademacher random vectors with entries in $\{\pm 1\}$. To compute the linear solves involving $(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}$, or $(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}$, respectively, in line 3, we again apply the preconditioned CG method. This algorithm also results in an unbiased and consistent approximation for $\mathrm{diag}(\boldsymbol{\Omega}_p)$, see Appendix C for a proof of Proposition 4.

Proposition 4 *Algorithm 2 produces an unbiased and consistent estimator of the predictive variance* $\mathit{diag}(\boldsymbol{\Omega}_p)$.

Both Algorithms 1 and 2 have a computational complexity of $\mathcal{O}\big((n + n_p) \cdot (\ell \cdot m + \ell \cdot m_v + m \cdot m_v)\big)$ for the calculation of predictive variances.

Algorithm 2 Stochastic Predictive Variance Estimator (SPV)

Require: Matrices $\boldsymbol{B}_{po}$, $\boldsymbol{D}_p$, $\boldsymbol{D}^{-1}$, $\boldsymbol{B}$, $\boldsymbol{B}_p^{-1}$, $\boldsymbol{\Sigma}_{mn}$, $\boldsymbol{\Sigma}_m$, $\boldsymbol{\Sigma}_{mn_p}$, $\boldsymbol{M}$, $\boldsymbol{W}$
Ensure: Approximated predictive variances $\mathrm{diag}(\boldsymbol{\Omega}_p)$
1: Initialize: $\boldsymbol{Z} \leftarrow \boldsymbol{0} \in \mathbb{R}^{n_p}$
2: **for** $i = 1$ to ℓ **do**
3: $\boldsymbol{z}_i^{(2)} \leftarrow \begin{cases} \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\boldsymbol{\Sigma}_\dagger^{-1}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1} \\ \cdot\boldsymbol{\Sigma}_\dagger^{-1}\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big)\boldsymbol{z}_i^{(1)} \\ \text{or } \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\boldsymbol{\Sigma}_\dagger^{-1}\boldsymbol{W}^{-1} \\ \cdot(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big)\boldsymbol{z}_i^{(1)} \end{cases}$
where $\boldsymbol{z}_i^{(1)} \sim$ Rademacher
4: $\boldsymbol{Z} \leftarrow \boldsymbol{Z} + \boldsymbol{z}_i^{(1)} \circ \boldsymbol{z}_i^{(2)}$
5: **end for**
6: $\mathrm{diag}(\boldsymbol{\Omega}_p) \leftarrow \mathrm{diag}(\boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T}) + \mathrm{diag}(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p})$
$- \mathrm{diag}(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}) + 2\cdot\mathrm{diag}(\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p})$
$- 2\cdot\mathrm{diag}(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}})$
$+ \mathrm{diag}(\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}})$
$+ \mathrm{diag}(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}) + \frac{1}{\ell}\boldsymbol{Z}$

Alternatively, Pleiss et al. (2018) propose to use the Lanczos algorithm to approximate predictive variances. However, recent research (Kündig and Sigrist, 2025; Gyger et al., 2026) has found that this Lanczos tridiagonalization-based approach is considerably more inaccurate compared to simulation-based methods. Intuitively, the Lanczos algorithm can work relatively well for approximating covariance matrices since their eigenvalue distribution often consists of a few large and many small eigenvalues, making them amenable to low-rank approximations. However, when applied to invert a covariance matrix, as required for computing predictive covariances, the eigenvalue distribution is reversed, resulting in many large eigenvalues and necessitating a very high rank for moderate accuracy.

4.3 Preconditioners

In the following, we present two preconditioners for iterative methods with VIF-Laplace approximations.

4.3.1 The VIF with Diagonal Update Preconditioner

The "**VIF** with **d**iagonal **u**pdate" (VIFDU) preconditioner is inspired by the "**V**ecchia **a**pproximation with **d**iagonal **u**pdate" (VADU) preconditioner introduced by Kündig and Sigrist (2025) and exploits the structure of the Vecchia approximation to approximate

$\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}+\boldsymbol{W} \approx \boldsymbol{B}^{\mathrm{T}}(\boldsymbol{D}^{-1}+\boldsymbol{W})\boldsymbol{B}$. Specifically, the VIFDU preconditioner $\widehat{\boldsymbol{P}}$ is given by

$$\begin{aligned}\widehat{\boldsymbol{P}} &= \boldsymbol{B}^{\mathrm{T}}\big(\boldsymbol{D}^{-1}-\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}+\boldsymbol{W}\big)\boldsymbol{B}\\ &\approx \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}-\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}+\boldsymbol{W}\\ &= (\boldsymbol{\Sigma}^{\mathrm{s}}+\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_{m}^{-1}\boldsymbol{\Sigma}_{mn})^{-1}+\boldsymbol{W}\\ &= \boldsymbol{\Sigma}_{\dagger}^{-1}+\boldsymbol{W}.\end{aligned}$$

This VIFDU preconditioner is applied when using the versions in (16) and (18) for the CG and SLQ methods, respectively. Linear solves with $\widehat{\boldsymbol{P}}$, the log-determinant $\log\det\big(\widehat{\boldsymbol{P}}\big)$ and its derivatives are computed in $\mathcal{O}\big(n\cdot(m_v\cdot m+m^2)\big)$ time. Sampling from $\mathcal{N}(\boldsymbol{0},\widehat{\boldsymbol{P}})$ is not straightforward since we can not apply the reparameterization trick used by Gardner et al. (2018a, Appendix C.1). If $\boldsymbol{\epsilon}_1\in\mathbb{R}^m$ and $\boldsymbol{\epsilon}_2\in\mathbb{R}^n$ are standard normal vectors, then $\big(\boldsymbol{B}^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{\frac{1}{2}}\boldsymbol{\epsilon}_2-\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-\frac{T}{2}}\boldsymbol{\epsilon}_1\big)$, is a sample from the distribution $\mathcal{N}\big(\boldsymbol{0},\boldsymbol{B}^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{D}^{-1}+\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1})\boldsymbol{B}\big)$ and not from $\mathcal{N}\big(\boldsymbol{0},\boldsymbol{B}^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{D}^{-1}-\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1})\boldsymbol{B}\big)$, where $\boldsymbol{M}^{\frac{1}{2}}$ is the Cholesky factor of $\boldsymbol{M}$ and $(\boldsymbol{W}+\boldsymbol{D}^{-1})^{\frac{1}{2}}$ is the elementwise square-root of $(\boldsymbol{W}+\boldsymbol{D}^{-1})$. However, sampling from $\mathcal{N}(\boldsymbol{0},\widehat{\boldsymbol{P}})$ can be done as follows. First, for sampling from $\mathcal{N}(\boldsymbol{0},\boldsymbol{\Sigma}_{\dagger}^{-1})$, we compute a sample from $\mathcal{N}(\boldsymbol{0},\boldsymbol{\Sigma}_{\dagger})$ as $\boldsymbol{B}^{-1}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{\epsilon}_2+\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_{m}^{-\frac{1}{2}}\boldsymbol{\epsilon}_1$, where $\boldsymbol{\Sigma}_{m}^{\frac{1}{2}}$ is the Cholesky factor of $\boldsymbol{\Sigma}_m$. Then we multiply by $\boldsymbol{\Sigma}_{\dagger}^{-1}$ to obtain a sample from $\mathcal{N}(\boldsymbol{0},\boldsymbol{\Sigma}_{\dagger}^{-1})$ and add $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}^{\frac{1}{2}}\boldsymbol{\epsilon}_3$, where $\boldsymbol{\epsilon}_3\in\mathbb{R}^n$ is a standard normal vector, such that $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}^{\frac{1}{2}}\boldsymbol{\epsilon}_3+\boldsymbol{\Sigma}_{\dagger}^{-1}(\boldsymbol{B}^{-1}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{\epsilon}_2+\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_{m}^{-\frac{1}{2}}\boldsymbol{\epsilon}_1)\sim\mathcal{N}\big(\boldsymbol{0},\boldsymbol{B}^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{D}^{-1}-\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1})\boldsymbol{B}\big)=\mathcal{N}(\boldsymbol{0},\widehat{\boldsymbol{P}})$. The computational overhead for this procedure is $\mathcal{O}\big(n\cdot(m_v+m)\big)$. For further information on the VIFDU preconditioner including how to do linear solves, calculate log-determinants and their derivatives, see Appendix E.1.

4.3.2 The FITC Preconditioner

The **FITC** preconditioner is given by

$$\widehat{\boldsymbol{P}}=\boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_{k}^{-1}\boldsymbol{\Sigma}_{kn}+\operatorname{diag}(\boldsymbol{\Sigma}-\boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_{k}^{-1}\boldsymbol{\Sigma}_{kn})+\boldsymbol{W}^{-1}$$

where $\boldsymbol{\Sigma}_k=\big[c_{\boldsymbol{\theta}}(\hat{\boldsymbol{s}}_i,\hat{\boldsymbol{s}}_j)\big]_{i=1:k,j=1:k}\in\mathbb{R}^{k\times k}$ and $\boldsymbol{\Sigma}_{kn}=\big[c_{\boldsymbol{\theta}}(\hat{\boldsymbol{s}}_i,\boldsymbol{s}_j)\big]_{i=1:k,j=1:n}\in\mathbb{R}^{k\times n}$ are (cross-) covariance matrices with respect to a set of inducing points $\widehat{\mathcal{S}}=\{\hat{\boldsymbol{s}}_1,\dots,\hat{\boldsymbol{s}}_k\}$. This preconditioner is applied when using the versions in (17) and (19) for the CG and SLQ methods, respectively. Note that the FITC preconditioner can use a different set of inducing points than those used in the VIF approximation. For example, using a larger number of inducing points can lead to a more effective preconditioner compared to relying solely on the inducing points used by the VIF approximation. The construction of the FITC preconditioner, linear solves involving $\widehat{\boldsymbol{P}}$, the computation of the log-determinant $\log\det\big(\widehat{\boldsymbol{P}}\big)$, and its derivatives all require $\mathcal{O}(n\cdot k^2)$ operations. Sampling from $\mathcal{N}(\boldsymbol{0},\widehat{\boldsymbol{P}})$ requires $\mathcal{O}(n\cdot k)$ operations. For additional details, see Appendix E.2.

An alternative low-rank preconditioner for Gaussian processes is the pivoted Cholesky decomposition (Harbrecht et al., 2012; Gardner et al., 2018a). However, the construction

of the pivoted Cholesky preconditioner is computationally more expensive than the FITC preconditioner. This means that the FITC preconditioner can use a higher rank for the same computational budget compared to the pivoted Cholesky preconditioner. Previous research (Gyger et al., 2026) has shown that the FITC preconditioner is more accurate than the pivoted Cholesky preconditioner, also for GP models that do not use inducing points.

5. Convergence analysis

In the following, we analyze the convergence properties of the preconditioned CG method with the VIFDU and FITC preconditioners. We show that the convergence speed is influenced by the VIF approximation tuning parameters, the number of inducing points m and the number of Vecchia neighbors m_v, and the eigenvalue structure of the covariance matrix $\boldsymbol{\Sigma}$. We denote the Frobenius and the 2-norm (spectral norm) by $||\cdot||_F$ and $||\cdot||_2$, respectively, and define the vector norm $||\boldsymbol{v}||_{\boldsymbol{A}} = \sqrt{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}}$ for $\boldsymbol{v} \in \mathbb{R}^n$ and a positive semi-definite matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. In the subsequent theorems, we make the following assumptions:

Assumption 1 *$n \geq 2$ and $m \in \{1, 2, \dots, n\}$.*

Assumption 2 *The covariance matrix $\boldsymbol{\Sigma}$ is of the form $\Sigma_{ij} = \sigma_1^2 \cdot r(\boldsymbol{s}_i, \boldsymbol{s}_j)$, where $r(\cdot,\cdot)$ is positive and continuous, and $r(\boldsymbol{s}_i, \boldsymbol{s}_i) = 1$. Additionally, the matrix $\boldsymbol{\Sigma}$ has eigenvalues $\lambda_1 \geq ... \geq \lambda_n > 0$.*

Assumption 3 *The response variable $\boldsymbol{y} \in \{0,1\}^n$ is binary, following a Bernoulli likelihood with a logit link function.*

Assumption 3 implies that the diagonal entries of $\boldsymbol{W}$, defined in equation (11), satisfy $0 \leq \boldsymbol{W}_{ii} \leq 1$ for $i \in \{1, 2, ..., n\}$.

First, we analyze the convergence speed of the preconditioned CG method for linear solves with $\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W}$ using the VIFDU preconditioner.

Theorem 5 ***Convergence of the CG method with the VIFDU preconditioner***

Let $\boldsymbol{\Sigma}_\dagger \in \mathbb{R}^{n\times n}$ be the VIF approximation of a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ with m inducing points and m_v Vecchia neighbors. Consider the linear system $(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\mathbf{u}^* = \mathbf{v}$, where $\mathbf{v} \in \mathbb{R}^n$. Let $\mathbf{u}_k$ be the approximation in the k-th iteration of the preconditioned CG method with the VIFDU preconditioner $\widehat{\boldsymbol{P}} \in \mathbb{R}^{n\times n}$. Under Assumptions 1–3, the following holds for the relative error:

$$\frac{\|\mathbf{u}^* - \mathbf{u}_k\|_{\boldsymbol{\Sigma}_\dagger^{-1}+\boldsymbol{W}}}{\|\mathbf{u}^* - \mathbf{u}_0\|_{\boldsymbol{\Sigma}_\dagger^{-1}+\boldsymbol{W}}} \leq 2 \cdot \left(\frac{1}{1 + \left(\alpha^n \cdot \left(\lambda_1 + (\lambda_{m+1} \cdot m_v)^n\right) \cdot \left(\sqrt{n} \cdot m_v + 1\right)\right)^{-1}} \right)^k,$$

where $\alpha > 1$ is a constant.

For the proof, see Appendix F. Theorem 5 shows that selecting less Vecchia neighbors m_v leads to faster convergence of the CG method. However, the relationship is more complicated regarding the number of inducing points m. While the term λ_{m+1} decreases with

larger m leading to faster convergence, we bound $||\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_{m}^{-1}\boldsymbol{\Sigma}_{mn}||_2$ in the proof of Theorem 5 by $||\boldsymbol{\Sigma}||_2 = \lambda_1$ (see Appendix F), and $||\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_{m}^{-1}\boldsymbol{\Sigma}_{mn}||_2$ grows with m. Additionally, we observe that the convergence is slower with increasing values of λ_1 and λ_{m+1}.

Next, we analyze the convergence speed of the preconditioned CG method for linear solves with $\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1}$ using the FITC preconditioner.

Theorem 6 ***Convergence of the CG method with the FITC preconditioner***

Let $\boldsymbol{\Sigma}_\dagger \in \mathbb{R}^{n\times n}$ be the VIF approximation of a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ with m inducing points and m_v Vecchia neighbors. Consider the linear system $(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\mathbf{u}^* = \mathbf{v}$, where $\mathbf{v} \in \mathbb{R}^n$. Let $\mathbf{u}_k$ be the approximation in the k-th iteration of the preconditioned CG method with the FITC preconditioner $\widehat{\boldsymbol{P}} \in \mathbb{R}^{n\times n}$. The preconditioner is constructed with the same set of inducing points as those used for $\boldsymbol{\Sigma}_\dagger$. Under Assumptions 1–3, the following holds for the relative error:

$$\frac{\|\mathbf{u}^* - \mathbf{u}_k\|_{\boldsymbol{\Sigma}_\dagger+\boldsymbol{W}^{-1}}}{\|\mathbf{u}^* - \mathbf{u}_0\|_{\boldsymbol{\Sigma}_\dagger+\boldsymbol{W}^{-1}}} \leq 2 \cdot \left(\frac{1}{1+(\alpha\cdot\lambda_{m+1}\cdot m_v)^{-n}}\right)^k,$$

where $\alpha > 1$ is a constant.

For the proof, see Appendix F. Theorem 6 shows that selecting fewer Vecchia neighbors m_v and a higher number of inducing points m leads to faster convergence of the preconditioned CG method. In contrast to the results in Theorem 5, the convergence rate of the CG method with the FITC preconditioner does not depend on the largest eigenvalue λ_1. Consequently, using the FITC preconditioner exhibits less sensitivity to the specific covariance matrix, the sample size, and the parametric covariance function.

6. Selecting inducing points and Vecchia neighbors

There are various ways of choosing inducing points. We use the kMeans++ algorithm (Arthur and Vassilvitskii, 2007) since Gyger et al. (2026) have found that it is an effective method for selecting inducing points for Gaussian processes with full-scale and FITC approximations. Its computational complexity is $\mathcal{O}(n\cdot m)$. If the covariance function is of the form

$$c_{\boldsymbol{\theta}}(\mathbf{s}_i, \mathbf{s}_j) = c^o_{\boldsymbol{\theta}}(\|q_{\boldsymbol{\lambda}}(\mathbf{s}_i) - q_{\boldsymbol{\lambda}}(\mathbf{s}_j)\|), \tag{22}$$

where $q_{\boldsymbol{\lambda}}(\cdot)$ is a transformation and $c^o_{\boldsymbol{\theta}}(\cdot)$ is an isotropic covariance function such as the Matérn kernel, then the distance $\|q_{\boldsymbol{\lambda}}(\mathbf{s}_i) - q_{\boldsymbol{\lambda}}(\mathbf{s}_j)\|$ depends on scaled inputs $\mathbf{s}^{\boldsymbol{\lambda}} = q_{\boldsymbol{\lambda}}(\mathbf{s})$. Automatic relevance determination (ARD) kernels with input dimension-specific length scales $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_d)$, $q_{\boldsymbol{\lambda}}(\mathbf{s}) = (s_1/\lambda_1, \dots, s_d/\lambda_d)$, are examples of such covariance functions. Since distances are computed in a transformed space, we also determine the inducing points using the transformed input locations $q_{\boldsymbol{\lambda}}(\mathbf{s})$. Intuitively, if a dimension s_l has a relatively large length scale λ_l for an ARD kernel, it is less relevant for the covariance function and should thus also have less impact on choosing inducing points. This is what is accomplished when choosing the inducing points using the transformed locations $q_{\boldsymbol{\lambda}}(\mathbf{s}) = (s_1/\lambda_1, \dots, s_d/\lambda_d)$. However, since the inducing points depend on $\boldsymbol{\lambda}$, the inducing points must be updated dynamically during the parameter optimization. An advantage of

the kMeans++ algorithm in terms of computational efficiency is that it can use the inducing points from a previous optimization iteration as initialization.

For selecting Vecchia neighbors, a common approach is to choose the m_v nearest input points that appear earlier in a given ordering of the data points. Usually, the Euclidean distance is used as a metric. However, previous research (Kang and Katzfuss, 2023) has shown that a correlation-based selection of neighbors can lead to more accurate approximations. For ARD covariance functions, for example, the intuitive argument why this choice of neighbors is generally beneficial is analogous to the one made above for choosing the inducing points in the transformed space: less relevant input dimensions should have less impact on the choice of neighbors. Often, correlation-based selection of neighbors is equivalent to simply using the Euclidean distance in a transformed space (Kang and Katzfuss, 2023). In our case, though, simply using the Euclidean metric in a transformed space is not applicable as we compute a Vecchia approximation for the residual process with covariance $\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}$, which corresponds to a non-stationary covariance function that cannot be written in the form of (22). In the following, we present a computationally efficient method for the correlation-based selection of Vecchia neighbors for VIF approximations.

We define the following correlation-based distance $d_c(\cdot,\cdot)$ to determine the nearest neighbors for the residual Vecchia approximation:

$$d_c(\mathbf{s}_i,\mathbf{s}_j) = \sqrt{1 - \left|\frac{c_r(\mathbf{s}_i,\mathbf{s}_j)}{\sqrt{c_r(\mathbf{s}_i,\mathbf{s}_i)\cdot c_r(\mathbf{s}_j,\mathbf{s}_j)}}\right|}, \quad c_r(\mathbf{s}_i,\mathbf{s}_j) = [\boldsymbol{\Sigma}]_{ij} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mj},$$

where $\boldsymbol{\Sigma}_{mj} = \left[c_{\boldsymbol{\theta}}(\boldsymbol{s}_i^*,\boldsymbol{s}_j)\right]_{i=1:m} \in \mathbb{R}^{m\times 1}$. There are efficient algorithms for finding the m_v nearest neighbors for the Euclidean distance, such as k-d trees and ball trees, enabling fast searches even in high-dimensional spaces. However, when employing an arbitrary metric, such as the proposed correlation-based metric, nearest neighbor search becomes significantly more challenging. Unlike the Euclidean distance, these specialized metrics often lack geometric properties that facilitate efficient search algorithms, leading to increased computational complexity. To address this, we utilize cover trees (Beygelzimer et al., 2006) which enable m_v-nearest neighbor search for a set of n points with a complexity of $\mathcal{O}\big(C_d \cdot n \cdot \log(m_v)\cdot(m_v+\log(n))\big)$, where $C_d > 0$ is a hidden dimensionality factor according to Beygelzimer et al. (2006).[2] For computational efficiency in our setting, we modify both the cover tree construction and the nearest neighbor search algorithm as follows. For the tree construction, instead of randomly selecting knots from the remaining data, we systematically insert the point with the smallest index into the cover tree. The full algorithm is given in Algorithm 3. This adaptation simplifies the nearest neighbor search by restricting potential neighbors to those with smaller indices, see Algorithm 4, resulting

2. Elkin and Kurlin (2023) identified potential issues in the original computational complexity analysis of Beygelzimer et al. (2006) and introduced new algorithms for constructing compressed cover trees and performing neighbor searches, which provably achieve the stated time complexity. Despite these refinements, the original and simpler algorithms proposed by Beygelzimer et al. (2006) have been shown to outperform other nearest neighbor search methods (Beygelzimer et al., 2006), and they are expected to achieve the claimed time complexity on well-behaved data sets in practice (Curtin, 2016). We use the original version of the cover tree algorithm throughout this paper, including for our computational complexity analysis. We conjecture that our adaptations could also be applied to the compressed versions, and we leave this as future work.

in improved runtime performance. Furthermore, we propose partitioning the data set into equally sized, sequentially ordered subsets, allowing for the parallel application of the cover tree algorithm, further reducing the effective runtime.

Algorithm 3 Construction of cover tree for correlation-based Vecchia neighbor search

Require: Set of data points $\mathcal{S} = \{\mathbf{s}_1, ..., \mathbf{s}_n\}$
Ensure: Cover tree $\mathcal{T}$
Initialize: Maximal distance: $R_{\max} = 1$
  Empty tree: $\mathcal{T}$
  Knot set of $\mathcal{T}$: $\mathcal{V} \leftarrow \{\}$
  Number of levels in $\mathcal{T}$: $l \leftarrow 0$
  Covered data points per level and knot: $\mathcal{C}_{0,0} \leftarrow \mathcal{S}$
Set root in $\mathcal{T}$: $\mathcal{T}_{1,1} \leftarrow \mathbf{s}_1$, $\mathcal{V} \leftarrow \{\mathbf{s}_1\}$, $l \leftarrow 1$, $\mathcal{C}_{l,\mathbf{s}_1} \leftarrow \mathcal{C}_{0,0} \setminus \{\mathbf{s}_1\}$
while size($\mathcal{V}$) $< n$ **do**
  $R_l \leftarrow R_{\max}/2^l$ ▷ New maximal distance between new knots in tree $\mathcal{T}$ and descendants
  $\mathcal{K} \leftarrow \mathcal{T}_{l,:}$ ▷ Set of knots of tree $\mathcal{T}$ in level l
  for $k \in \mathcal{K}$ **do**
    ind $\leftarrow 0$ ▷ Index of knot in tree $\mathcal{T}$ at level $l+1$
    while $\mathcal{C}_{l,k} \neq \{\}$ **do**
      ind $\leftarrow$ ind $+ 1$
      $\mathcal{T}_{l+1,\text{ind}} \leftarrow \{\mathbf{s}_i \in \mathcal{C}_{l,k} \mid i = \min\{j \mid \mathbf{s}_j \in \mathcal{C}_{l,k}\}\}$ ▷ Point with smallest index in $\mathcal{C}_{l,k}$
      $\mathcal{V} \leftarrow \mathcal{V} \cup \{\mathcal{T}_{l+1,\text{ind}}\}$
      $\mathcal{C}_{l,k} \leftarrow \mathcal{C}_{l,k} \setminus \{\mathcal{T}_{l+1,\text{ind}}\}$
      $\mathcal{C}_{l+1,\mathcal{T}_{l+1,\text{ind}}} \leftarrow \{\mathbf{s}_i \in \mathcal{C}_{l,k} \mid d_c(\mathbf{s}_i, \mathcal{T}_{l+1,\text{ind}}) \leq R_l\}$
      $\mathcal{C}_{l,k} \leftarrow \mathcal{C}_{l,k} \setminus \mathcal{C}_{l+1,\mathcal{T}_{l+1,\text{ind}}}$
    end while
  end for
  $l \leftarrow l + 1$
end while

Similarly as we re-determine the inducing points during parameter estimation, we also update the correlation-based Vecchia neighbors. In our subsequent experiments, we re-determine both the inducing points and the Vecchia neighbors in every optimization iteration that is a power of two. Additionally, we update the inducing points and Vecchia neighbors when the optimization has converged and restart the optimization if the log-marginal likelihood changes after this update, and repeat this process until convergence is achieved.

7. Simulated experiments

We first analyze our methods through experiments on simulated data. We generate samples from zero-mean Gaussian processes with d-dimensional inputs and automatic relevance determination (ARD) covariance functions, a marginal variance of 1, and length scale pa-

Algorithm 4 Find m_v nearest Vecchia neighbors with respect to the metric d_c

Require: Query point $\mathbf{s}_i \in \mathcal{S}$, cover tree $\mathcal{T}$, number of neighbors m_v
Ensure: $\mathcal{N}_{m_v}$ set of m_v nearest Vecchia neighbors of $\mathbf{s}_i$
Initialize: Maximal distance: $R_{\max} = 1$
Number of levels in tree $\mathcal{T}$: $l \leftarrow \text{depth}(\mathcal{T})$
Set of potential nearest neighbors: $\mathcal{Q} \leftarrow \{\mathcal{T}_{1,1}\}$
Distance to m_v closest point in $\mathcal{Q}$: $D_{m_v} \leftarrow 1$ ▷ Set to 1 if $|\mathcal{Q}| < m_v$
for $j = 1$ to l **do**
  $\mathcal{C} \leftarrow \{\mathbf{s}_k \in \text{Children}(\mathbf{s}) \mid \mathbf{s} \in \mathcal{Q} \cap k < i\} \cup \mathcal{Q}$ ▷ Children of $\mathcal{Q}$ with index $< i$
  $D_{m_v} \leftarrow m_v$-th smallest distance $d_c(\mathbf{s}, \mathbf{s}_i)$ for $\mathbf{s} \in \mathcal{C}$ ▷ Set to 1 if $|\mathcal{C}| < m_v$
  $\mathcal{Q} \leftarrow \{\mathbf{s} \in \mathcal{C} \mid d_c(\mathbf{s}, \mathbf{s}_i) \leq D_{m_v} + 1/2^{j-1}\}$
end for
$\mathcal{N}_{m_v} \leftarrow$ Find m_v nearest points to $\mathbf{s}_i$ in $\mathcal{Q}$ by brute-force search

rameters $\boldsymbol{\lambda} \in \mathbb{R}^d$. See the following subsections for the specific choices of $\boldsymbol{\lambda}$ and the sample size n. Sample locations are drawn uniformly from the unit hypercube $[0,1]^d$. We simulate data for both Gaussian and non-Gaussian likelihoods. For the Gaussian likelihood, we use a variance of 0.001 for the error term, and in the non-Gaussian case, the response variable $\boldsymbol{y} \in \{0,1\}^n$ follows a Bernoulli likelihood with a logit link function. Unless stated otherwise, we use $m = 200$ inducing points and $m_v = 30$ Vecchia neighbors for VIF, Vecchia, and FITC approximations, but other choices are also considered below. Except as otherwise indicated, Vecchia neighbors for VIF and Vecchia approximations are selected using the correlation-based distance described in Section 6, and inducing points for VIF and FITC approximations are selected in the transformed space as outlined in Section 6. Furthermore, unless specified otherwise, we use the FITC preconditioner with $k = 200$ inducing points for the iterative methods for VIF approximations, Algorithm 1 (SBPV) with $\ell = 100$ sample vectors for predictive variances, 50 sample vectors for the SLQ method, and a convergence tolerance of $\delta = 0.01$ for the CG method. Estimation is done by minimizing the negative log-marginal likelihood using the limited-memory BFGS algorithm. All calculations are performed on an AMD EPYC 7742 processor with 64 CPU cores and 512 GB of RAM, using the `GPBoost` library version 1.5.8 compiled with GCC 11.2.0. To ensure reproducibility, the code for the experiments is available at `https://github.com/TimGyger/VIF`.

7.1 Accuracy for varying input dimensions and smoothness of covariance functions

We begin by comparing the accuracy of VIF approximations with FITC and Vecchia standalone approximations for varying input dimensions and covariance functions with different smoothness parameters. Specifically, we consider input dimensions $d \in \{2, 5, 10, 20, 50, 100\}$ and the following ARD covariance functions: 1/2-Matérn, 3/2-Matérn, 5/2-Matérn, and Gaussian (∞-Matérn). Since these covariance functions have different effective ranges and the average distance among two randomly chosen points changes with the dimension d, we adapt the data-generating length scale parameters for different input dimensions d and covariance functions for better comparability. See Table 5 in Appendix 7 for the specific

length scale parameters. Standalone Vecchia approximations are applied to the observable response variable. We have also run the experiments applying a Vecchia approximation to the latent GP and obtained almost equal prediction accuracy measures (results not shown). For each setting, we generate 10 independent data sets consisting of 20,000 samples using a Gaussian likelihood. To assess prediction accuracy, we randomly select 10,000 samples from each data set to serve as test data. Prediction accuracy is measured using the RMSE, the log-score (LS), $-\frac{1}{n_p}\sum_{i=1}^{n_p}\log\left(\phi(\frac{y_i^*-\mu_{\dagger,i}^p}{\sigma_{\dagger,i}^p})\right)$, and the continuous ranked probability score (CRPS) $-\frac{1}{n_p}\sum_{i=1}^{n_p}\sigma_{\dagger,i}^p(\frac{1}{\sqrt{\pi}}-2\cdot\phi(\frac{y_i^*-\mu_{\dagger,i}^p}{\sigma_{\dagger,i}^p})-\frac{y_i^*-\mu_{\dagger,i}^p}{\sigma_{\dagger,i}^p}(2\cdot\Phi(\frac{y_i^*-\mu_{\dagger,i}^p}{\sigma_{\dagger,i}^p})-1))$, where $\phi(x)=\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right)$ and $\Phi(x)$ are the density and cumulative distribution functions, respectively, of a standard normal distribution, y^* is the test response, and $\mu_{\dagger,i}^p$ and $\sigma_{\dagger,i}^p$ are the predictive means and variances, respectively.

Figure 2 compares the VIF, FITC, and Vecchia approximations and an exact GP model without an approximation for varying input dimensions d for a 3/2-Matérn kernel. We observe that the VIF approximation consistently outperforms both the Vecchia and FITC approximations across all input dimensions. As expected, for low-dimensional inputs, the Vecchia approximation is very accurate, and the FITC approximation is substantially less accurate. However, the accuracy of the Vecchia approximation declines relatively quickly with increasing dimension d, and the FITC approximation is considerably more accurate for large dimensions. Moreover, for higher dimensions $d \geq 10$, none of the approximations achieves the accuracy of an exact GP model, highlighting that modeling high-dimensional functions is challenging. We also repeat these experiments for a different choice of parameters. Specifically, in Figure 14 in Appendix 7, we present the results for a 3/2-Matérn kernel using a larger error variance of $\sigma^2 = 0.01$, and the length scale parameters are chosen such that the covariance remains approximately equal at the average distance for all d (to the one of a Gaussian kernel with length scales $\boldsymbol{\lambda} = (0.35, 0.4, 0.45, 0.5, 0.55)^{\mathrm{T}}$). The results are similar to those shown in Figure 2.

In Table 4 in Appendix 7, we additionally report the corresponding average runtimes for training and prediction for each method. As expected, the training runtimes for ARD kernels increase with the input dimension d across all methods as the corresponding optimization problems become more high-dimensional. Overall, VIF and FITC approximations have similar runtimes for estimation, and Vecchia approximations are faster. All three methods have similar prediction runtimes. As mentioned above, we use $m = 200$ inducing points and $m_v = 30$ Vecchia neighbors for VIF, Vecchia, and FITC approximations. In Figure 10 in Appendix 7, we additionally show results obtained by increasing the number of Vecchia neighbors to $m_v = 60$ for the Vecchia approximation to ensure that the computational time for estimation is approximately equal for all three methods. Although the Vecchia approximation slightly benefits from the increased approximation complexity, the results are qualitatively very similar. Moreover, Figures 11 and 12 in Appendix 7 compare the prediction accuracy versus the total training and prediction runtime for the three approximations across varying numbers of inducing points and Vecchia neighbors, for input dimensions $d = 10$ and $d = 100$. For $d = 100$, the VIF approximation achieves the best accuracy-runtime trade-off, while for $d = 10$, Vecchia and VIF approximations exhibit similar accuracy-runtime trade-offs. Furthermore, we also compare the VIF approximation for two inducing-point-to-neighbor ratios m/m_v in Figures 11 and 12 in Appendix 7. In line

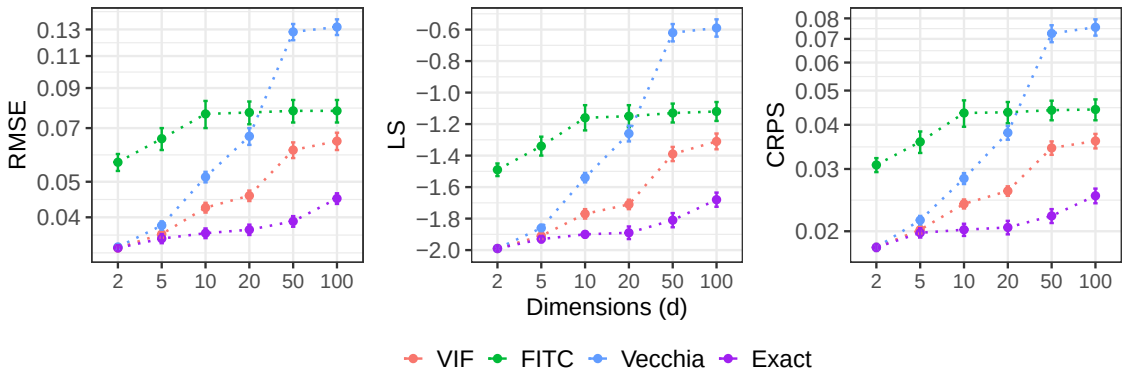

Figure 2: Comparison of the VIF, FITC, and Vecchia approximations and an exact GP model (without an approximation) for varying dimensions d using an ARD 3/2-Matérn kernel. The plots show the RMSE (log-scale), log-score (LS), and CRPS (log-scale) (mean $\pm$ 2 standard errors).

with the above findings, these results indicate that in high dimensions, it is beneficial for the VIF approximations to rather increase the number of inducing points than the number of Vecchia neighbors.

The results when comparing VIF, FITC, and Vecchia approximations and an exact GP model without an approximation for the different covariance functions with varying smoothness and $d = 10$ are shown in Figure 3. The VIF approximation is again consistently more accurate compared to both FITC and Vecchia approximations. All approximations are more accurate for smoother covariance functions. However, the relative difference in accuracy between the Vecchia and both the VIF and FITC approximations increases with increasing smoothness levels. This is in line with the prior expectation that Vecchia approximations are more accurate for moderately smooth covariance functions, whereas FITC approximations become more accurate for smoother covariance functions. In Figure 13 in Appendix 7, we additionally report the results for varying smoothness when $d = 2$, i.e., for spatial data. As expected, the VIF approximation outperforms the Vecchia approximation only marginally in this low-dimensional setting. For the Gaussian kernel, all approximations are almost equally accurate.

7.2 Comparison of preconditioners

For all subsequent experiments and unless stated otherwise, we generate 100,000 samples from a zero-mean Gaussian process with five-dimensional inputs and an ARD Gaussian covariance function with length scale parameters $\boldsymbol{\lambda} = (0.15, 0.30, 0.45, 0.60, 0.75)$.

In the following, we analyze the accuracy and runtimes of the iterative methods for non-Gaussian likelihoods introduced in Section 4. First, we compare the VIFDU and FITC preconditioners introduced in Section 4.3 with respect to the runtime and accuracy of log-likelihood approximations for a binary likelihood. In Figure 4, we report the wall-clock times and the RMSE between log-marginal likelihoods computed using iterative methods

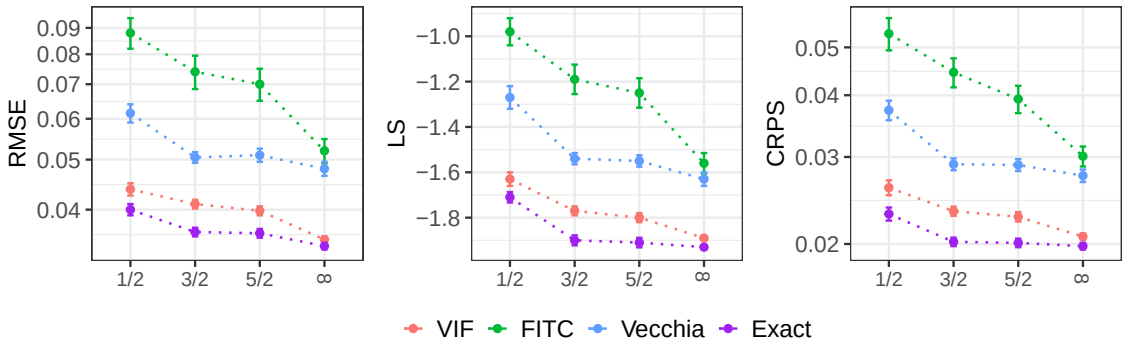

Figure 3: Comparison of the VIF, FITC, and Vecchia approximations and an exact GP model (without an approximation) for varying smoothness parameters of an ARD Matérn kernel for $d = 10$ (1/2-Matérn, 3/2-Matérn, 5/2-Matérn, and ∞-Matérn = Gaussian kernel). The plots show the RMSE (log-scale), log-score (LS), and CRPS (log-scale) (mean $\pm$ 2 standard errors).

and those computed using a Cholesky decomposition for three different VIF approximations with different numbers of inducing points m and Vecchia neighbors m_v. The figure also shows the runtime for the calculations based on a Cholesky decomposition. The marginal likelihood is evaluated at the data-generating parameters and repeated 100 times. Overall, both preconditioners lead to very accurate approximations. For instance, a log-marginal likelihood difference in 10 corresponds to a relative error of approximately 10^{-4}. However, we find that the FITC preconditioner consistently outperforms the VIFDU preconditioner, having both faster runtimes and more accurate log-marginal likelihood approximations. Moreover, the iterative methods are substantially faster than traditional Cholesky-based computations with a speed-up of approximately three orders of magnitude for both preconditioners. Note that both preconditioners yield unbiased stochastic log-likelihood approximations (results not shown).

The accuracy and runtimes of the iterative methods depend on the CG convergence tolerance δ and the number of sample vectors ℓ for the SLQ method. Table 6 in Appendix 7 reports the accuracy and runtimes for different tolerances $\delta \in \{0.0001, 0.001, 0.01, 0.1, 1\}$ and numbers of sample vectors $\ell \in \{10, 50, 100\}$. The results show that decreasing δ below 0.01 yields negligible improvements in accuracy while increasing computational cost, indicating that $\delta = 0.01$ provides a good balance between accuracy and efficiency. In contrast, the number of sample vectors ℓ has a more pronounced impact on the accuracy of SLQ-approximated log-determinants than the CG tolerance.

The FITC preconditioner has one tuning parameter, the number of inducing points, which governs a trade-off between computational efficiency and the accuracy of the SLQ approximation. To analyze this parameter, we compute the log-marginal likelihood using varying numbers of inducing points, $k \in \{10, 50, 100, 200, 300, 400, 500\}$. In Figure 15 in Appendix 7, we report the runtime and the log-marginal likelihood compared to exact Cholesky-based calculations for three different VIF approximations. Our analysis shows

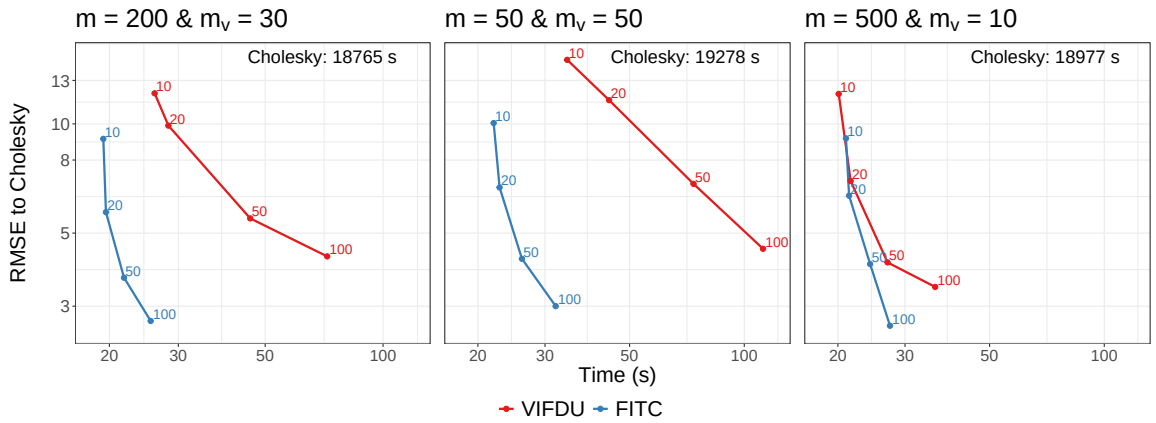

Figure 4: Accuracy-runtime comparison of preconditioners: RMSE between log-marginal likelihoods computed using iterative methods and those computed using a Cholesky decomposition versus runtime for the VIFDU and FITC preconditioners and varying numbers of sample vectors ℓ (annotated in the plot). A binary likelihood, $n = 100{,}000$ samples, and three different VIF approximations are used.

that the FITC preconditioner overall yields the fastest runtime for $k = 200$. Note that the experiments in this and the next section are conducted on only one simulated data set, since we do not want to mix sampling variability and randomness of the methods that are analyzed. However, the results do not change when using other samples (results not shown).

7.3 Predictive variances

Next, we compare the two simulation- and iterative-methods-based approaches introduced in Section 4.2 for calculating predictive variances, namely Algorithm 1 (SBPV) and Algorithm 2 (SPV). Both algorithms are run twice with the VIFDU and the FITC preconditioners. We use simulated data with $n = n_p = 100{,}000$ training and test points, a binary likelihood for the response variable, and predictive distributions are calculated using the true data-generating covariance parameters. To measure the accuracy of the methods, we compute the RMSE of the simulation-based predictive variances relative to the "exact" Cholesky-based results. In addition, we measure the total runtime, which includes computing the mode, the latent predictive means, and the latent predictive variances. Figure 5 shows the RMSE as a function of the wall-clock time for varying numbers of random vectors ℓ. We find that all algorithms yield very accurate predictive variances already for small runtimes. Furthermore, the results show that the SBPV algorithm is more accurate than the SPV algorithm, and that the FITC preconditioner consistently leads to lower runtimes compared to the VIFDU preconditioner.

7.4 Runtime analysis of VIF approximations

In the following, we analyze how the runtime of a VIF approximation scales with the sample size, and how it depends on the two tuning parameters of a VIF approximation: the number

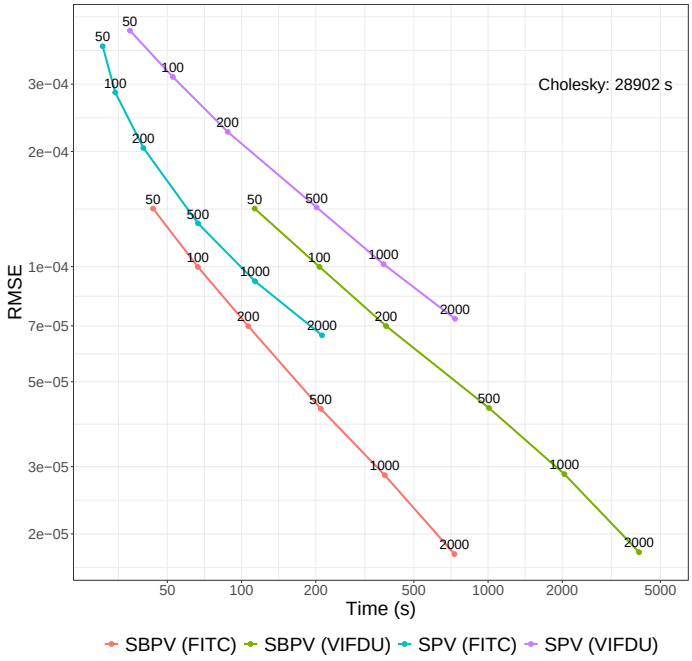

Figure 5: Accuracy-runtime comparison of simulation- and iterative-methods-based predictive variances (SBPV and SPV algorithms with the FITC and VIFDU preconditioners): RMSE between predictive variances computed using simulation-based methods and those computed using a Cholesky decomposition versus runtime for a binary likelihood. The number of random vectors ℓ is annotated in the plot.

of inducing points m and the number of Vecchia neighbors m_v. Specifically, we vary each of the three quantities m, m_v, and n in turn while keeping the others fixed at $m = 200$, $m_v = 30$, and $n = 100{,}000$. We consider both a Gaussian and a Bernoulli likelihood, and we additionally compare VIF approximations to the FITC and Vecchia approximations. For the Bernoulli likelihood, we apply the VIF approximations using our proposed iterative methods with the FITC and VIFDU preconditioners. Similarly, the standalone Vecchia approximation uses iterative methods with the Vecchia approximation with diagonal update (VADU) preconditioner (Kündig and Sigrist, 2025). Figure 6 illustrates the runtime for evaluating the log-marginal likelihood for Gaussian and Bernoulli likelihoods for different approximation parameter values $m \in \{10, 20, 50, 100, 200, 500\}$, $m_v \in \{5, 10, 15, 20, 30, 50\}$, and sample sizes $n \in \{10{,}000, 20{,}000, 30{,}000, 50{,}000, 80{,}000, 100{,}000\}$. As expected, the runtime increases with the sample size n, the number of inducing points m, and the number of Vecchia neighbors m_v for both likelihoods. For the non-Gaussian likelihood, we find that the FITC preconditioner consistently outperforms the VIFDU preconditioner across all parameter settings. Notably, the VIF approximation with the FITC preconditioner has similar runtimes as the Vecchia approximation.

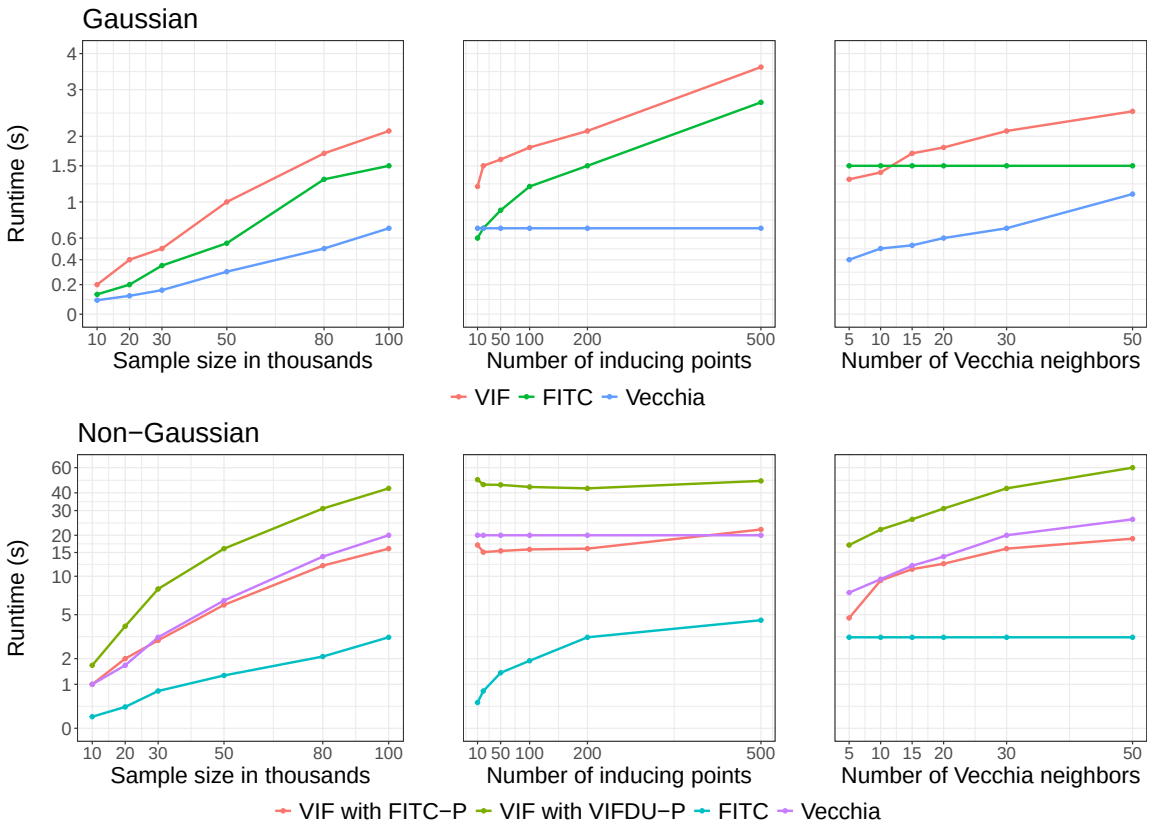

Figure 6: Time (s) for computing the marginal likelihood with VIF, FITC, and Vecchia approximations for varying sample sizes n, numbers of inducing points m, and numbers of Vecchia neighbors m_v for both a Gaussian (top row) and a non-Gaussian likelihood (bottom row).

For the experiments presented in Figure 6, we exclude the time required to determine the m_v Vecchia neighbors, as these are not necessarily recomputed for each log-likelihood evaluation in the optimization process. Figure 7 illustrates the runtime for constructing the cover tree and identifying the m_v nearest neighbors with respect to the correlation distance for varying sample sizes, feature dimensions $d \in \{2, 5, 10, 20, 50, 100\}$, numbers of inducing points, and number of Vecchia neighbors. We again vary each of these quantities in turn while keeping the others fixed at $m = 200$, $m_v = 30$, $n = 100{,}000$, and $d = 5$. These results show that the runtime primarily depends on the sample size n and the input dimension d. The runtime scales approximately linearly with the number of inducing points, which is expected, as computing the correlation distance requires $\mathcal{O}(m)$ operations. The number of Vecchia neighbors m_v has only a minor impact on the overall runtime.

In Figure 16 in Appendix 7, we additionally analyze the runtime required to compute the predictive means and variances for Gaussian and Bernoulli likelihoods, respectively, for different approximation parameters, sample sizes, number of prediction points, and comparing VIF, FITC, and Vecchia approximations. For the VIF approximation and the non-Gaussian

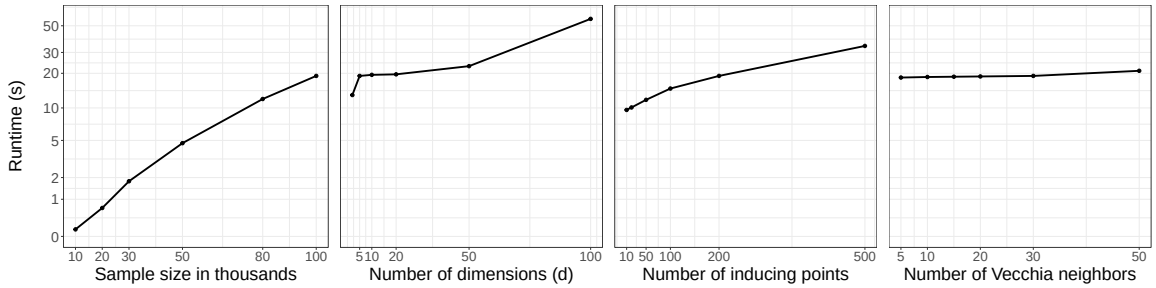

Figure 7: Time (s) for constructing the cover tree and finding the m_v nearest neighbors with respect to the correlation distance for varying sample sizes n, input dimensions d, numbers of inducing points m, and numbers of Vecchia neighbors m_v.

likelihood, iterative methods are employed with the FITC and VIFDU preconditioners. As the results show, the FITC preconditioner consistently outperforms the VIFDU preconditioner across all parameter settings and even surpasses the Vecchia approximation with the VADU preconditioner in terms of runtime.

8. Real-world data experiments

In the following, we apply our proposed methods to multiple real-world data sets comparing the VIF approximation to the following state-of-the-art GP approximations: sparse Gaussian process regression (SGPR) (Titsias, 2009), stochastic variational Gaussian processes (SVGP) (Hensman et al., 2013, 2015), scalable kernel interpolation for product kernels (SKIP) (Gardner et al., 2018b), and double-Kullback-Leibler-optimal Gaussian process approximation (DKLGP) (Cao et al., 2023). SGPR and SVGP are both variational low-rank inducing-point approximations which differ in the way the evidence lower bound (ELBO) is maximized. SKIP, an extension of structured kernel interpolation (SKI) (Wilson and Nickisch, 2015), is an inducing points-based approximation that relies on local kernel interpolation, the fast Fourier transform (FFT), and iterative methods. DKLGP uses a sparse inverse Cholesky factor approximation whose parameters are trained in a variational framework. In other words, DKLGP is a variational form of a Vecchia approximation.

We use `GPyTorch` (Gardner et al., 2018a) version 1.13, for SKIP, SGPR, and SVGP. For DKLGP, we use the implementation of Cao et al. (2023) available in the repository https://github.com/katzfuss-group/DKL-GP. Furthermore, for SKIP, SGPR, SVGP, and DKLGP, training is done using the ADAM optimizer, following the recommendations of Gardner et al. (2018a) and Cao et al. (2023). For SVGP, the inducing points are optimized jointly with the variational parameters, and for SGPR, the inducing points are chosen by random subsampling from the training data. Unless otherwise specified, a 3/2-Matérn ARD kernel is used. For the non-Gaussian data sets, we employ the VIF approximation with iterative methods and the same settings as in the simulated experiments; see the beginning of Section 7. We use 5-fold cross-validation to assess the prediction accuracy. All data sets and code to reproduce our experiments are available on `https://github.com/TimGyger/VIF`.

8.1 Data sets

We consider a variety of data sets from the UCI and OpenML repositories, which are widely used for benchmarking GP approximations in the machine learning literature, as well as additional spatial data sets. Appropriate likelihoods are selected based on the characteristics of each data set; see below for the specific choices. We follow Cao et al. (2023) and apply the following pre-processing steps. First, the input features are transformed to the interval $[0, 1]$ based on the training data. Second, input features with a standard deviation below 0.01 after standardization are excluded from the analysis to prevent numerical issues for some of the methods. For the same reason, input features with a standard deviation below 0.1 are additionally excluded for the binary classification data sets. Third, the data sets are filtered to ensure a minimum Euclidean distance $\|\mathbf{s}_i - \mathbf{s}_j\|$ of 0.001 between any two data points, retaining only the first point in cases where multiple points are too close, to prevent numerical singularities for some methods. With the exception of the 3dRoad data set, this procedure has a negligible impact on the sample size. For the data sets modeled using a Gaussian likelihood, the response variable is normalized based on the training data sets to have zero mean and unit variance for better comparability across different data sets.

8.2 Results

We first present the results for the Gaussian likelihood in Table 1. Entries marked as NA indicate numerical problems (SKIP did not converge and yields predictions that are very inaccurate). Additionally, we illustrate the log-score (LS) results in Figure 8 (left plot). We find that the VIF approximation consistently outperforms all other methods in terms of all accuracy measures across all data sets. Furthermore, for data sets with low input dimensions ('3dRoad', 'Protein', 'Kin40K'), we observe that the inducing point methods SKIP, SGPR, and SVGP are very inaccurate, and the variational Vecchia approximation DKLGP is more accurate compared to these inducing point-based methods. The situation is reversed for the higher-dimensional data sets ('KEGGU', 'KEGG', 'Elevators', 'Ailerons'), where the best inducing-points approximation, SVGP, is more accurate than the variational Vecchia approximation DKLGP. These findings are in line with the results from the simulated experiments in Section 7.1, where we found that Vecchia approximations are accurate for low-dimensional inputs but less accurate for higher dimensions. The VIF approximation achieves the highest prediction accuracy for all data sets with both low and higher-dimensional inputs as it combines low-rank inducing points and residual process Vecchia approximations. Concerning computational time, we find that the VIF approximation is, overall, faster than all other approximations.

Next, we present the results for the binary data sets in Table 2 reporting the area under the receiver operating characteristic curve (AUC), the root mean squared error (RMSE), which in this case corresponds to the square root of the Brier score, the accuracy (ACC), and the log-score (LS). Figure 8 (right plot) additionally visualizes the log-score (LS) for all data sets modeled using non-Gaussian likelihoods. The differences in prediction accuracy among all approximations are very small for the binary classification data sets. The likely explanation for this is that binary response variable data provide less information compared to continuous and count data response variables. Furthermore, the VIF approximation has

Data	Accuracy measure	**VIF** $m_v = 30$ $m = 200$	**SKIP** $m = 1000$	**SGPR** $m = 1000$	**SVGP** $m = 1000$	**DKLGP** $\rho = 1.5$
3dRoad $n = 434{,}874$ $d = 3$	RMSE	**0.145 ± 0.002**	0.603 ± 0.010	0.734 ± 0.014	0.550 ± 0.008	0.255 ± 0.004
	CRPS	**0.068 ± 0.002**	0.294 ± 0.014	0.402 ± 0.008	0.297 ± 0.004	0.121 ± 0.002
	LS	**-0.625 ± 0.010**	2.371 ± 0.022	1.114 ± 0.020	0.825 ± 0.006	0.009 ± 0.004
	Time	376 s	1861 s	759 s	966 s	890 s
KEGGU $n = 63{,}608$ $d = 26$	RMSE	**0.094 ± 0.008**	NA convergence issues	0.117 ± 0.024	**0.094 ± 0.004**	0.146 ± 0.014
	CRPS	**0.030 ± 0.002**		0.068 ± 0.008	0.034 ± 0.002	0.054 ± 0.002
	LS	**-1.081 ± 0.072**		-0.441 ± 0.081	-0.955 ± 0.042	-0.845 ± 0.038
	Time	738 s		502 s	617 s	1362 s
KEGG $n = 48{,}827$ $d = 18$	RMSE	**0.100 ± 0.010**	0.672 ± 0.166	0.699 ± 0.170	**0.103 ± 0.004**	0.142 ± 0.008
	CRPS	**0.041 ± 0.004**	0.407 ± 0.086	0.450 ± 0.088	0.048 ± 0.002	0.067 ± 0.002
	LS	**-1.082 ± 0.070**	1.158 ± 0.134	1.194 ± 0.144	-0.893 ± 0.024	-0.593 ± 0.040
	Time	383 s	1202 s	863 s	951 s	1858 s
Elevators $n = 16{,}599$ $d = 17$	RMSE	**0.355 ± 0.004**	NA convergence issues	0.692 ± 0.224	0.373 ± 0.008	0.418 ± 0.006
	CRPS	**0.196 ± 0.004**		0.458 ± 0.202	0.205 ± 0.004	0.230 ± 0.004
	LS	**0.388 ± 0.012**		2.045 ± 1.534	0.435 ± 0.020	0.907 ± 0.054
	Time	711 s		520 s	1195 s	2583 s
Protein $n = 45{,}730$ $d = 8$	RMSE	**0.516 ± 0.006**	0.831 ± 0.008	0.822 ± 0.048	0.729 ± 0.006	0.621 ± 0.006
	CRPS	**0.259 ± 0.002**	0.610 ± 0.008	0.467 ± 0.028	0.411 ± 0.004	0.327 ± 0.006
	LS	**0.667 ± 0.016**	6.310 ± 0.216	1.234 ± 0.058	1.105 ± 0.008	0.900 ± 0.052
	Time	547 s	1807 s	676 s	1309 s	1531 s
Kin40K $n = 40{,}000$ $d = 8$	RMSE	**0.114 ± 0.002**	1.401 ± 0.024	0.321 ± 0.008	0.175 ± 0.004	0.284 ± 0.002
	CRPS	**0.059 ± 0.002**	0.862 ± 0.020	0.271 ± 0.010	0.099 ± 0.002	0.151 ± 0.002
	LS	**-0.808 ± 0.012**	3.192 ± 0.034	0.131 ± 0.008	-0.230 ± 0.012	0.082 ± 0.008
	Time	333 s	1328 s	947 s	1179 s	2732 s
Ailerons $n = 13{,}750$ $d = 33$	RMSE	**0.399 ± 0.019**	NA convergence issues	0.428 ± 0.062	**0.386 ± 0.015**	0.424 ± 0.028
	CRPS	**0.208 ± 0.008**		0.238 ± 0.043	**0.208 ± 0.003**	0.230 ± 0.009
	LS	**0.436 ± 0.030**		0.676 ± 0.251	**0.459 ± 0.017**	0.642 ± 0.068
	Time	528 s		209 s	510 s	1918 s

Table 1: RMSE, CRPS, log-score (LS) (mean ± 2 standard errors), and runtime in seconds (s) for the regression data sets modeled using a Gaussian likelihood. The score of the best-performing method is shown in bold. If the mean score of another method lies within two standard errors of the best mean, it is also in bold.

a faster runtime than the most accurate inducing points method (SVGP) and the variational Vecchia approximation DKLGP.

In Table 3, we present the results for data sets modeled using Poisson, Student-t, and Gamma likelihoods. Specifically, the House data set is modeled with a Student-t likelihood, the Bike data set with a Poisson likelihood, and the Power and WaterVapor (WV) data sets with a Gamma likelihood. The current implementation of DKLGP by Cao et al. (2023) available on https://github.com/katzfuss-group/DKL-GP/ does not support Poisson or Gamma likelihoods. The NA entry indicates that SGPR crashed. We find that the VIF

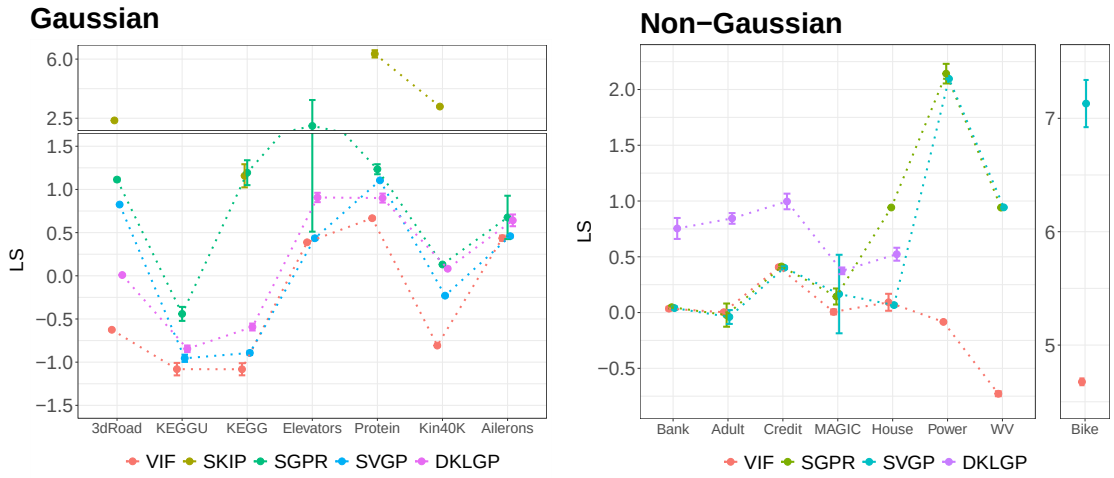

Figure 8: Log-score (LS) with error bars (mean ± 2 standard errors) for the data sets modeled with a Gaussian (left plot) and non-Gaussian likelihoods (right plot).

approximation consistently outperforms all other methods in prediction accuracy across all data sets. The strong performance of the VIF-Laplace approximation compared to the variational benchmark methods is likely due to its more accurate covariance approximation. Concerning computational time, the VIF approximation has a similar runtime as SVGP for two data sets ('Power', 'WaterVapor'), but it is slower on the two other data sets ('Bike', 'House').

In Tables 8 and 9 in Appendix 7, we additionally report results comparing the VIF approximation to classical Vecchia and FITC approximations. Overall, the VIF approximation is more accurate than both Vecchia and FITC approximations. The Vecchia approximation also yields competitive prediction accuracy, often approaching that of the VIF approximation. However, the VIF approximation achieves higher accuracy than the Vecchia approximations on multiple data sets such as the 'Kin40K', 'Elevators', and 'Ailerons' data sets.

8.3 Matérn kernel smoothness selection and non-zero mean functions

Assuming a fixed value for the smoothness parameter ν of the Matérn covariance function and a zero prior mean function $F(\cdot) = 0$ might result in misspecified models. In the following, we analyze these two issues. We first extend our comparison of VIF approximations for the regression tasks with Gaussian and non-Gaussian likelihoods from the previous section by additionally estimating the shape (or smoothness) parameter ν of the Matérn ARD kernel, rather than assuming a fixed $\nu = 3/2$ Matérn ARD kernel. This approach enables model selection within the Matérn kernel family. Note that estimating the smoothness parameter is possible for VIF approximations implemented in the `GPBoost` library but not for the other approximations implemented in `GPyTorch`. The results are shown in Figure 9 (left plot) for the log-score and in more detail in Table 7 in Appendix 7. We find that estimating ν improves the prediction accuracy across nearly all data sets, but the differences

Data	Accuracy measure	**VIF** $m_v = 30$ $m = 200$	**SGPR** $m = 1000$	**SVGP** $m = 1000$	**DKLGP** $\rho = 1.5$
Bank $n = 45{,}211$ $d = 16$	AUC	**0.806 ± 0.008**	**0.800 ± 0.006**	**0.804 ± 0.008**	0.789 ± 0.006
	RMSE	**0.284 ± 0.004**	**0.286 ± 0.004**	**0.284 ± 0.002**	0.292 ± 0.004
	ACC	**0.895 ± 0.002**	**0.896 ± 0.002**	**0.896 ± 0.002**	0.892 ± 0.004
	LS	**0.035 ± 0.016**	**0.048 ± 0.012**	**0.040 ± 0.014**	0.754 ± 0.094
	Time	1643 s	1270 s	2603 s	5012 s
Adult $n = 48{,}842$ $d = 14$	AUC	**0.880 ± 0.008**	**0.887 ± 0.007**	**0.883 ± 0.006**	0.869 ± 0.004
	RMSE	**0.336 ± 0.006**	**0.332 ± 0.004**	**0.335 ± 0.004**	0.342 ± 0.004
	ACC	**0.838 ± 0.006**	**0.840 ± 0.007**	**0.835 ± 0.006**	0.833 ± 0.004
	LS	**0.003 ± 0.012**	**-0.023 ± 0.104**	**-0.040 ± 0.062**	0.844 ± 0.048
	Time	2854 s	1561 s	5661 s	2815 s
Credit $n = 30{,}000$ $d = 22$	AUC	0.768 ± 0.006	0.767 ± 0.006	**0.773 ± 0.004**	0.752 ± 0.004
	RMSE	**0.378 ± 0.004**	**0.379 ± 0.004**	**0.377 ± 0.004**	0.385 ± 0.004
	ACC	**0.808 ± 0.006**	**0.808 ± 0.006**	**0.809 ± 0.006**	0.799 ± 0.006
	LS	**0.407 ± 0.012**	**0.413 ± 0.012**	**0.402 ± 0.012**	0.996 ± 0.070
	Time	1161 s	1235 s	2786 s	2816 s
MAGIC $n = 19{,}020$ $d = 9$	AUC	**0.920 ± 0.004**	0.902 ± 0.006	0.915 ± 0.004	0.900 ± 0.004
	RMSE	**0.316 ± 0.004**	0.334 ± 0.004	0.321 ± 0.004	0.340 ± 0.004
	ACC	**0.866 ± 0.006**	0.850 ± 0.006	**0.860 ± 0.006**	0.842 ± 0.004
	LS	**0.006 ± 0.022**	0.144 ± 0.072	0.166 ± 0.352	0.374 ± 0.030
	Time	236 s	437 s	663 s	459 s

Table 2: AUC, RMSE (Brier score), ACC, LS (mean ± 2 standard errors), and runtime in seconds (s) for the binary classification data sets. Bold indicates the best mean; other means are in bold if within two standard errors.

are relatively small for most data sets except for the '3dRoad', 'Kin40K', 'House', 'Power', and 'Ailerons' data sets, for which estimated ν's yield substantially higher accuracy. In addition, estimating the smoothness parameter increases the runtime due to the additional evaluations of Bessel functions.

Next, we extend the GP model (1) by allowing for non-zero prior mean (fixed effects) functions $F(\cdot)$. Specifically, we consider a linear regression function $F(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{\beta}$ as well as a function that is modeled using tree-boosting (Sigrist, 2022a,b). The latter is denoted as 'GPBoost model' in the following. In both cases, we use the GP input variables $\boldsymbol{s}$ as covariates $\boldsymbol{x}$ for the mean function, i.e., $\boldsymbol{s} = \boldsymbol{x}$. In addition, we use a VIF approximation and a 3/2-Matérn kernel. The results are shown in Figure 9 (right plot) for the log-score and in more detail in Table 10 in Appendix 7. We find that while the GPBoost model yields slightly more accurate predictions in terms of the log-score for some data sets ('Elevators', 'Protein', 'Kin40K'), all three models have overall similar prediction accuracy measures except for the linear model on the 'KEGGU' data set.

Data	Accuracy measure	**VIF** $m_v = 30$ $m = 200$	**SGPR** $m = 1000$	**SVGP** $m = 1000$	**DKLGP** $\rho = 1.5$
Bike $n = 17{,}379$ $d = 12$ (Poisson)	RMSE CRPS LS Time	**38.904 ± 1.332** **17.162 ± 0.454** **4.676 ± 0.030** 1534 s	NA crashed	42.256 ± 0.850 20.746 ± 0.194 7.130 ± 0.208 817 s	not implemented
House $n = 20{,}640$ $d = 8$ (Student-t)	RMSE CRPS LS Time	**0.214 ± 0.008** **0.103 ± 0.002** 0.092 ± 0.076 1932 s	0.271 ± 0.004 0.259 ± 0.002 0.942 ± 0.002 349 s	0.258 ± 0.002 0.137 ± 0.002 **0.067 ± 0.012** 568 s	0.267 ± 0.004 0.144 ± 0.002 0.523 ± 0.058 525 s
Power $n = 52{,}417$ $d = 5$ (Gamma)	RMSE CRPS LS Time	**0.218 ± 0.002** **0.121 ± 0.001** **-0.084 ± 0.010** 5035 s	0.826 ± 0.456 0.837 ± 0.090 2.142 ± 0.088 1597 s	0.567 ± 0.030 0.787 ± 0.006 2.096 ± 0.004 4063 s	not implemented
WaterVapor $n = 100{,}000$ $d = 2$ (Gamma)	RMSE CRPS LS Time	**0.102 ± 0.001** **0.056 ± 0.001** **-0.728 ± 0.022** 3975 s	0.423 ± 0.002 0.293 ± 0.001 0.941 ± 0.001 2027 s	0.424 ± 0.004 0.294 ± 0.002 0.943 ± 0.004 4747 s	not implemented

Table 3: RMSE, CRPS, log-score (LS) (mean ± 2 standard errors), and runtime in seconds (s) for the regression data sets modeled using non-Gaussian likelihoods. Bold indicates the best mean; other means are in bold if within two standard errors.

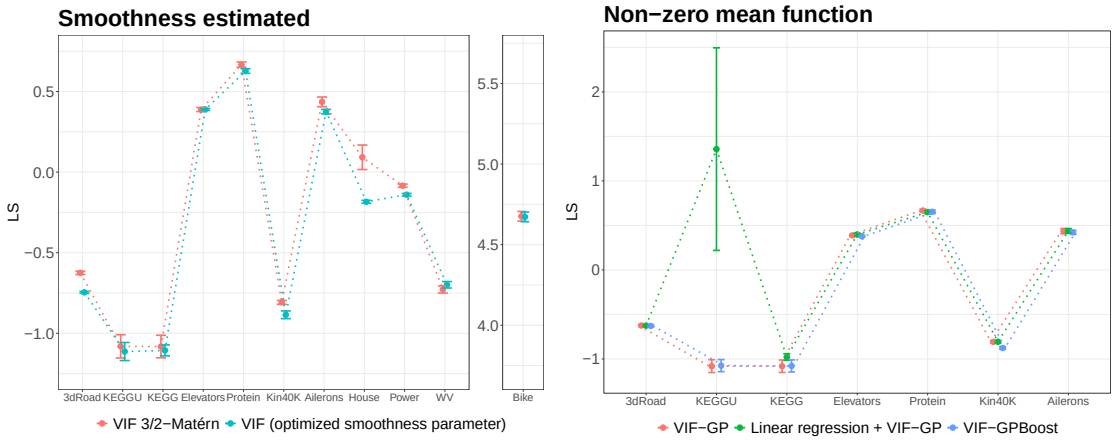

Figure 9: Log-score (LS) with error bars (mean ± 2 standard errors) when estimating the smoothness parameter for the regression data sets (left plot) and when using non-zero prior mean functions (right plot).

9. Conclusion

In this work, we introduce Vecchia-inducing-points full-scale (VIF) approximations, a novel approach that leverages the complementary strengths of global inducing points and local Vecchia approximations to enhance the scalability and accuracy of Gaussian processes. We develop an efficient correlation-based neighbor selection strategy for the residual process using a modified cover tree algorithm, and we introduce iterative methods and preconditioners for VIF-Laplace approximation for non-Gaussian likelihoods, significantly reducing computational costs by several orders of magnitude compared to Cholesky-based computations. Our theoretical analysis provides insights into the convergence properties of the preconditioned conjugate gradient method. We systematically evaluate our methods concerning both runtime and accuracy through extensive numerical experiments on simulated and real-world data sets. We find that VIF approximations consistently outperform state-of-the-art alternatives in both accuracy and computational efficiency across various real-world data sets. Future research can analyze alternative methods for choosing both inducing points and Vecchia neighbors in VIF approximations, e.g., using a variational framework.

Acknowledgments

This research was partially supported by the Swiss Innovation Agency - Innosuisse (grant number '57667.1 IP-ICT').

Appendix

Appendix A. Gradients of the residual process Vecchia approximation

The gradients of $\boldsymbol{B}$ and $\boldsymbol{D}$ defined in Section 2.1 with respect to the covariance parameters $\boldsymbol{\theta}$ are given by

$$\begin{aligned}
\left(\frac{\partial \boldsymbol{B}}{\partial \boldsymbol{\theta}}\right)_{iN(i)} = &-\Big(\frac{\partial \boldsymbol{\Sigma}_{iN(i)}}{\partial \boldsymbol{\theta}} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_{mN(i)}}{\partial \boldsymbol{\theta}} - \frac{\partial \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)} \\
&+ \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_m}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\Big)\big(\boldsymbol{\Sigma}_{N(i)} - \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\big)^{-1} \\
&+ \big(\boldsymbol{\Sigma}_{iN(i)} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\big)\big(\boldsymbol{\Sigma}_{N(i)} - \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\big)^{-1} \\
&\cdot \Big(\frac{\partial \boldsymbol{\Sigma}_{N(i)}}{\partial \boldsymbol{\theta}} - \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_{mN(i)}}{\partial \boldsymbol{\theta}} - \frac{\partial \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)} \\
&+ \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_m}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\Big)\big(\boldsymbol{\Sigma}_{N(i)} - \boldsymbol{\Sigma}_{mN(i)}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mN(i)}\big)^{-1} \\
\frac{\partial \boldsymbol{D}}{\partial \boldsymbol{\theta}} = &\frac{\partial \boldsymbol{\Sigma}_i}{\partial \boldsymbol{\theta}} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_{mi}}{\partial \boldsymbol{\theta}} - \frac{\partial \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi} + \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_m}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi} \\
&- \frac{\partial \boldsymbol{A}_i}{\partial \boldsymbol{\theta}}\big(\boldsymbol{\Sigma}_i - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi}\big) \\
&- \boldsymbol{A}_i\Big(\frac{\partial \boldsymbol{\Sigma}_i}{\partial \boldsymbol{\theta}} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_{mi}}{\partial \boldsymbol{\theta}} - \frac{\partial \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi} + \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_m}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi}\Big)
\end{aligned}$$

and cost also $\mathcal{O}\big(n \cdot (m_v^3 + m_v^2 \cdot m)\big)$.

Appendix B. VIF and Laplace Approximation

Log-determinant:

$$\begin{aligned}
\log\det(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \boldsymbol{I}_n) &= \log\det(\boldsymbol{\Sigma}_\dagger) + \log\det(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1}) \\
&= \log\det(\boldsymbol{M}) - \log\det(\boldsymbol{\Sigma}_m) - \log\det(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}) + \log\det(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1}) \\
&= \log\det(\boldsymbol{M}) - \log\det(\boldsymbol{\Sigma}_m) - \log\det(\boldsymbol{D}^{-1}) \\
&\quad + \log\det(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}) \\
&= \log\det(\boldsymbol{M}) - \log\det(\boldsymbol{\Sigma}_m) - \log\det(\boldsymbol{D}^{-1}) \\
&\quad + \log\det(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}) - \log\det(\boldsymbol{M}) + \log\det\big(\boldsymbol{M}_1\big) \\
&= -\log\det(\boldsymbol{\Sigma}_m) - \log\det(\boldsymbol{D}^{-1}) + \log\det(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}) \\
&\quad + \log\det\big(\boldsymbol{M}_1\big),
\end{aligned}$$

where $\boldsymbol{M}_1 = \boldsymbol{M} - \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}$.

Linear solves:

$$
\begin{aligned}
(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1} &= \boldsymbol{W}^{-1}(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\boldsymbol{\Sigma}_\dagger = \boldsymbol{W}^{-1}(\boldsymbol{W}^{-1}+\boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T}+\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn})^{-1}\boldsymbol{\Sigma}_\dagger \\
&= \boldsymbol{W}^{-1}\big((\boldsymbol{W}^{-1}+\boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1} \\
&\quad -(\boldsymbol{W}^{-1}+\boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_2^{-1}\boldsymbol{\Sigma}_{mn}(\boldsymbol{W}^{-1}+\boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\big)\boldsymbol{\Sigma}_\dagger \\
&= \boldsymbol{W}^{-1}\big(\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W} \\
&\quad -\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B} \\
&\quad \cdot(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\big)\boldsymbol{\Sigma}_\dagger,
\end{aligned}
$$

where $\boldsymbol{M}_2 = \boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_{mn}(\boldsymbol{W}^{-1}+\boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}$ and $\boldsymbol{M}_3 = \boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}$.

Gradients:

The gradients with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\xi}$ are given by

$$
\begin{aligned}
\frac{\partial L^{VIFLA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})}{\partial\theta_k} =& \frac{1}{2}\tilde{\boldsymbol{b}}^T\frac{\partial\boldsymbol{\Sigma}_\dagger^{-1}}{\partial\theta_k}\tilde{\boldsymbol{b}} + \frac{1}{2}\frac{\partial\log\det(\boldsymbol{\Sigma}_\dagger\boldsymbol{W}+\boldsymbol{I}_n)}{\partial\theta_k} \\
&+\left(\frac{\partial L^{VIFLA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})}{\partial\tilde{\boldsymbol{b}}}\right)^{\mathrm{T}}\frac{\partial\tilde{\boldsymbol{b}}}{\partial\theta_k}, \quad k=1,\dots,q, \\
\frac{\partial L^{VIFLA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})}{\partial\xi_l} =& -\frac{\partial\log p(\boldsymbol{y}\mid\boldsymbol{b}^*,\boldsymbol{\xi})}{\partial\xi_l} + \frac{1}{2}\frac{\partial\log\det(\boldsymbol{\Sigma}_\dagger\boldsymbol{W}+\boldsymbol{I}_n)}{\partial\xi_l} \\
&+\left(\frac{\partial L^{LA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})}{\partial\tilde{\boldsymbol{b}}}\right)^{\mathrm{T}}\frac{\partial\tilde{\boldsymbol{b}}}{\partial\xi_l}, \quad l=1,\dots,r,
\end{aligned}
$$

where

$$
\begin{aligned}
\frac{\partial L^{VIFLA}(\boldsymbol{y},\boldsymbol{\theta},\boldsymbol{\xi})}{\partial b^*(\boldsymbol{s}_i)} &= \frac{1}{2}\frac{\partial\log\det(\boldsymbol{\Sigma}_\dagger\boldsymbol{W}+\boldsymbol{I}_n)}{\partial b^*(\boldsymbol{s}_i)} = \frac{1}{2}\operatorname{Tr}\left((\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\frac{\partial\boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}\right) \\
&\quad +\frac{1}{2}\operatorname{Tr}\Big(\boldsymbol{M}_1^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} \\
&\quad \cdot\frac{\partial\boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\Big),
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial\tilde{\boldsymbol{b}}}{\partial F_i} &= -\left(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1}\right)^{-1}\boldsymbol{W}_{\cdot i} = -\boldsymbol{W}^{-1}\big(\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W} \\
&\quad -\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \\
&\quad \cdot\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\big)\boldsymbol{\Sigma}_\dagger\boldsymbol{W}_{\cdot i}, \\
\frac{\partial\tilde{\boldsymbol{b}}}{\partial\theta_k} &= -\left(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1}\right)^{-1}\frac{\partial\boldsymbol{\Sigma}_\dagger^{-1}}{\partial\theta_k}\tilde{\boldsymbol{b}} = -\boldsymbol{W}^{-1}\big(\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W} \\
&\quad -\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \\
&\quad \cdot\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\big)\boldsymbol{\Sigma}_\dagger\frac{\partial\boldsymbol{\Sigma}_\dagger^{-1}}{\partial\theta_k}\tilde{\boldsymbol{b}},
\end{aligned}
$$

$$\begin{aligned}\frac{\partial \tilde{\boldsymbol{b}}}{\partial \xi_l} &= \left(\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1}\right)^{-1} \frac{\partial^2 \log p\left(\boldsymbol{y} \mid \boldsymbol{b}^*, \boldsymbol{\xi}\right)}{\partial \xi_l \partial \tilde{\boldsymbol{b}}} \\ &= \boldsymbol{W}^{-1}\big(\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W} - \boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \\ &\quad \cdot \boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^T\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}\big)\boldsymbol{\Sigma}_\dagger \frac{\partial^2 \log p\left(\boldsymbol{y} \mid \boldsymbol{b}^*, \boldsymbol{\xi}\right)}{\partial \xi_l \partial \tilde{\boldsymbol{b}}},\end{aligned}$$

$$\begin{aligned}\frac{\partial \log\det\left(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \boldsymbol{I}_n\right)}{\partial b^*(\boldsymbol{s}_i)} &= \frac{\partial\big(\log\det(\boldsymbol{\Sigma}_\dagger) + \log\det(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})\big)}{\partial b^*(\boldsymbol{s}_i)} = \frac{\partial\big(\log\det(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})\big)}{\partial b^*(\boldsymbol{s}_i)} \\ &= \operatorname{Tr}\left((\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\frac{\partial \boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}\right) \\ &\quad + \operatorname{Tr}\Big(\boldsymbol{M}_1^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} \\ &\quad \cdot \frac{\partial \boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\Big) \quad i=1,\dots,p,\end{aligned}$$

$$\begin{aligned}\frac{\partial \log\det\left(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \boldsymbol{I}_n\right)}{\partial \theta_k} &= -\operatorname{Tr}\left(\boldsymbol{\Sigma}_m^{-1}\frac{\partial \boldsymbol{\Sigma}_m}{\partial \theta_k}\right) + \operatorname{Tr}\left(\boldsymbol{M}_1^{-1}\frac{\partial \boldsymbol{M}_1}{\partial \theta_k}\right) + \operatorname{Tr}\left(\boldsymbol{D}^{-1}\frac{\partial \boldsymbol{D}}{\partial \theta_k}\right) \\ &\quad + \operatorname{Tr}\left((\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\frac{\partial(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})}{\partial \theta_k}\right) \quad k=1,\dots,q, \\ \frac{\partial \log\det\left(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \boldsymbol{I}_n\right)}{\partial \xi_l} &= \operatorname{Tr}\left((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial \boldsymbol{W}}{\partial \xi_l}\right) \quad l=1,\dots,r.\end{aligned}$$

Appendix C. Predictive (co-)variances

C.1 Gaussian

$$\begin{aligned}\boldsymbol{\Sigma}_\dagger^p &= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\ &\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\widetilde{\boldsymbol{\Sigma}}_\dagger^{-1}\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\ &\quad - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \\ &= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\ &\quad + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\ &\quad - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\ &\quad - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\ &\quad + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\end{aligned}$$

C.2 Non-Gaussian

$$
\begin{aligned}
\boldsymbol{\Omega}_p &= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\big(\boldsymbol{\Sigma}_\dagger^{-1} - \boldsymbol{\Sigma}_\dagger^{-1}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\boldsymbol{\Sigma}_\dagger^{-1}\big) \\
&\quad \cdot \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \\
&\overset{(1)}{=} \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \\
&= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big) \\
&\quad \cdot \big((\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1} \\
&\quad - (\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_2^{-1}\boldsymbol{\Sigma}_{mn}(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\big) \\
&\quad \cdot \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \\
&= \boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad - \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1} \\
&\quad \cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \qquad (23) \\
&\quad + \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big) \\
&\quad \cdot(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_2^{-1}\boldsymbol{\Sigma}_{mn}(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1} \qquad (24) \\
&\quad \cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big),
\end{aligned}
$$

where in (1) we used the Sherman-Woodbury-Morrison formula. We compute the terms in (23) and (24) separately:

$$
\begin{aligned}
&\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1} \\
&\quad\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \\
&\overset{(1)}{=} \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big) \\
&\quad\cdot\big(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\big) \\
&\quad\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big) \\
&= \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad -\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\
&\quad -\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\
&\quad -\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} \\
&\quad +\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}} \\
&\quad +\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}(\boldsymbol{W}+\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p},
\end{aligned}
$$

where in (1) we used again the Sherman-Woodbury-Morrison formula.

$$
\begin{aligned}
&\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\\
&\quad\cdot(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_2^{-1}\boldsymbol{\Sigma}_{mn}(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1}\\
&\quad\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big)\\
&\overset{(2)}{=}\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\big)\\
&\quad\cdot\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\\
&\quad\cdot\big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\big)\\
&= \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\\
&\quad+\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\\
&\quad\cdot\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\\
&\quad\cdot\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&\quad-\boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}\\
&\quad\cdot(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&\quad-\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}\\
&\quad\cdot(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}},
\end{aligned}
$$

where in (2) we used that $(\boldsymbol{W}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{D}\boldsymbol{B}^{-T})^{-1} = \boldsymbol{W}^{-1}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}$. Finally, we obtain:

$$
\begin{aligned}
\boldsymbol{\Omega}_p = &\boldsymbol{B}_p^{-1}\boldsymbol{D}_p\boldsymbol{B}_p^{-T} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} - \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&+ \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p} + \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\\
&+ \boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\\
&+ \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&- \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\\
&- \boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&+ \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}\\
&+ \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\\
&\cdot \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}\\
&\cdot (\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\\
&\cdot \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&- \boldsymbol{B}_p^{-1}\boldsymbol{B}_{po}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}\\
&\cdot (\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn_p}\\
&- \boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}(\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{W}^{-1}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{W}^{-1}\\
&\cdot (\boldsymbol{W} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}_{po}^{\mathrm{T}}\boldsymbol{B}_p^{-\mathrm{T}}.
\end{aligned}
$$

While Algorithms 1 and 2 produce estimators based on sampled realizations, the following proofs are formulated in terms of the underlying random vectors $\boldsymbol{z}_i$.

Proof [Proof of Proposition 3] By standard results, in Algorithm 1, the only non-deterministic term $1/\ell \sum_{i=1}^{\ell} \boldsymbol{z}_i^{(8)} (\boldsymbol{z}_i^{(8)})^{\mathrm{T}}$ is an unbiased and consistent estimator for $\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1}\big) \boldsymbol{\Sigma}_\dagger^{-1} \big(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W}\big)^{-1} \boldsymbol{\Sigma}_\dagger^{-1} \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1} \boldsymbol{B}_{po}^{\mathrm{T}} \boldsymbol{B}_p^{-\mathrm{T}}\big)$, and the claim in Proposition 3 thus follows. ■

Proof [Proof of Proposition 4] By standard results, in Algorithm 2, the only non-deterministic term $1/\ell \sum_{i=1}^{\ell} \boldsymbol{z}_i^{(1)} \circ \big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1}\big) \boldsymbol{\Sigma}_\dagger^{-1} (\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})^{-1} \boldsymbol{\Sigma}_\dagger^{-1} \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1} \boldsymbol{B}_{po}^{\mathrm{T}} \boldsymbol{B}_p^{-\mathrm{T}}\big) \boldsymbol{z}_i^{(1)}$ is an unbiased and consistent estimate of $\operatorname{diag}\big(\boldsymbol{\Sigma}_{mn_p}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn} - \boldsymbol{B}_p^{-1} \boldsymbol{B}_{po} (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1}\big) \boldsymbol{\Sigma}_\dagger^{-1} (\boldsymbol{W} + \boldsymbol{\Sigma}_\dagger^{-1})^{-1} \boldsymbol{\Sigma}_\dagger^{-1} \big(\boldsymbol{\Sigma}_{mn}^{\mathrm{T}} \boldsymbol{\Sigma}_m^{-1} \boldsymbol{\Sigma}_{mn_p} - (\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}^{-1} \boldsymbol{B})^{-1} \boldsymbol{B}_{po}^{\mathrm{T}} \boldsymbol{B}_p^{-\mathrm{T}}\big)$, and the claim in Proposition 4 thus follows. ■

Appendix D. Derivations of SLQ-approximated log-determinants and stochastic trace estimation for gradients

The stochastic Lanczos quadrature (SLQ) method can be derived as follows:

$$\begin{aligned}
\log\det\big(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \mathbf{I}_n\big) &= \log\det\big(\boldsymbol{\Sigma}_\dagger\big) + \log\det\big(\boldsymbol{W} + \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1}\big) \\
&= \log\det\big(\boldsymbol{\Sigma}_\dagger\big) + \log\det\big(\boldsymbol{P}^{-\frac{1}{2}} (\boldsymbol{W} + \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1}) \boldsymbol{P}^{-\frac{1}{2}}\big) + \log\det(\boldsymbol{P}) \\
&\approx \log\det\big(\boldsymbol{\Sigma}_\dagger\big) + \frac{1}{\ell} \sum_{i=1}^{\ell} \boldsymbol{z}_i^{\mathrm{T}} \boldsymbol{P}^{-\frac{1}{2}} \widetilde{\boldsymbol{Q}}_i \log\big(\widetilde{\boldsymbol{T}}_i\big) \widetilde{\boldsymbol{Q}}_i^{\mathrm{T}} \boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{z}_i + \log\det(\boldsymbol{P}) \\
&\approx \log\det\big(\boldsymbol{\Sigma}_\dagger\big) + \frac{n}{\ell} \sum_{i=1}^{\ell} \boldsymbol{e}_1^{\mathrm{T}} \log\big(\widetilde{\boldsymbol{T}}_i\big) \boldsymbol{e}_1 + \log\det(\boldsymbol{P}),
\end{aligned}$$

where $\widetilde{\boldsymbol{Q}}_i \in \mathbb{R}^{n \times k}$ has orthonormal columns, $\widetilde{\boldsymbol{T}}_i \in \mathbb{R}^{k \times k}$ is the tridiagonal matrix of the partial Lanczos tridiagonalization $\widetilde{\boldsymbol{Q}}_i \widetilde{\boldsymbol{T}}_i \widetilde{\boldsymbol{Q}}_i^{\mathrm{T}} \approx \boldsymbol{P}^{-\frac{1}{2}} (\boldsymbol{W} + \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1}) \boldsymbol{P}^{-\frac{1}{2}}$ obtained by running the Lanczos algorithm for k steps with $\boldsymbol{P}^{-\frac{1}{2}} (\boldsymbol{W} + \widetilde{\boldsymbol{\Sigma}}_\dagger^{-1}) \boldsymbol{P}^{-\frac{1}{2}}$ and initial vector $\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{z}_i / \sqrt{n} \approx \boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{z}_i / \|\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{z}_i\|_2$, and $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{P})$. Alternatively, we have

$$\begin{aligned}
\log\det\big(\boldsymbol{\Sigma}_\dagger \boldsymbol{W} + \mathbf{I}_n\big) &= \log\det\big(\boldsymbol{W}\big) + \log\det\big(\boldsymbol{W}^{-1} + \widetilde{\boldsymbol{\Sigma}}_\dagger\big) = \log\det\big(\boldsymbol{W}\big) \\
&\quad + \log\det\big(\boldsymbol{W}^{-1} + \widetilde{\boldsymbol{\Sigma}}_\dagger\big) \\
&\approx \log\det\big(\boldsymbol{W}\big) + \frac{n}{\ell} \sum_{i=1}^{\ell} \boldsymbol{e}_1^{\mathrm{T}} \log\big(\widetilde{\boldsymbol{T}}_i\big) \boldsymbol{e}_1 + \log\det(\boldsymbol{P}),
\end{aligned}$$

where $\widetilde{\boldsymbol{T}}_i \in \mathbb{R}^{k \times k}$ is the tridiagonal matrix of the partial Lanczos tridiagonalization $\widetilde{\boldsymbol{Q}}_i \widetilde{\boldsymbol{T}}_i \widetilde{\boldsymbol{Q}}_i^{\mathrm{T}} \approx \boldsymbol{P}^{-\frac{1}{2}} (\boldsymbol{W}^{-1} + \widetilde{\boldsymbol{\Sigma}}_\dagger) \boldsymbol{P}^{-\frac{1}{2}}$ obtained by running the Lanczos algorithm for k steps with $\boldsymbol{P}^{-\frac{1}{2}} (\boldsymbol{W}^{-1} + \widetilde{\boldsymbol{\Sigma}}_\dagger) \boldsymbol{P}^{-\frac{1}{2}}$ and initial vector $\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{z}_i / \|\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{z}_i\|_2$, and $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{P})$.

Furthermore, the stochastic trace estimation for gradients can be computed as follows:

$$\begin{aligned}
\mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial b^*(\boldsymbol{s}_i)}\Big) &= \mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial b^*(\boldsymbol{s}_i)}\mathbb{E}_{\boldsymbol{z}_i\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{P})}\left[\boldsymbol{P}^{-1}\boldsymbol{z}_i\boldsymbol{z}_i^{\mathrm{T}}\right]\Big)\\
&\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial\boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\\
&= \widetilde{T}_\ell^{b^*(\boldsymbol{s}_i)} \quad i=1,\ldots,p,\\
\mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\theta_k}\Big) &\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\\
&= \widetilde{T}_\ell^{\theta_k} \quad k=1,\ldots,q,\\
\mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\xi_l}\Big) &\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial\boldsymbol{W}}{\partial\xi_l}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\\
&= \widetilde{T}_\ell^{\xi_l} \quad l=1,\ldots,r,
\end{aligned}$$

or e.g.

$$\begin{aligned}
\mathrm{Tr}\Big((\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\frac{\partial(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)}{\partial\theta_k}\Big) &\approx \frac{1}{\ell}\sum_{i=1}^{\ell}\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger^{-1})\big)\big(\frac{\partial(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\\
&= \widetilde{T}_\ell^{\theta_k} \quad k=1,\ldots,q.
\end{aligned}$$

Further, we can reduce the variance of this stochastic estimate by using the preconditioner $\boldsymbol{P}$ to build a control variate, e.g.,

$$\begin{aligned}
\mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\theta_k}\Big) \approx \frac{1}{\ell}\sum_{i=1}^{\ell}\Big(&\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\big(\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\\
&-\hat{c}_{opt}\cdot\big(\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{P}^{-1}\big)\big(\frac{\partial\boldsymbol{P}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\Big)+\hat{c}_{opt}\cdot\mathrm{Tr}\Big(\boldsymbol{P}^{-1}\frac{\partial\boldsymbol{P}}{\partial\theta_k}\Big),
\end{aligned}$$

$$\hat{c}_{\mathrm{opt}}=\frac{\sum_{i=1}^{\ell}\Big(\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})^{-1}\big)\Big(\frac{\partial(\boldsymbol{W}+\boldsymbol{\Sigma}_\dagger^{-1})}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\Big)-\widetilde{T}_\ell\Big)\Big(\big(\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{P}^{-1}\big)\Big(\frac{\partial\boldsymbol{P}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\Big)-\mathrm{Tr}\Big(\boldsymbol{P}^{-1}\frac{\partial\boldsymbol{P}}{\partial\theta_k}\Big)\Big)}{\sum_{i=1}^{\ell}\Big[\big(\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{P}^{-1}\big)\Big(\frac{\partial\boldsymbol{P}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\Big)-\mathrm{Tr}\Big(\boldsymbol{P}^{-1}\frac{\partial\boldsymbol{P}}{\partial\theta_k}\Big)\Big]^2}.$$

Or,

$$\begin{aligned}
\mathrm{Tr}\Big((\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\frac{\partial(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)}{\partial\theta_k}\Big) \approx \frac{1}{\ell}\sum_{i=1}^{\ell}\Big(&\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\big)\big(\frac{\partial(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\\
&-\hat{c}_{opt}\cdot\big(\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{P}^{-1}\big)\big(\frac{\partial\boldsymbol{P}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\big)\Big)+\hat{c}_{opt}\cdot\mathrm{Tr}\Big(\boldsymbol{P}^{-1}\frac{\partial\boldsymbol{P}}{\partial\theta_k}\Big),
\end{aligned}$$

$$\hat{c}_{\mathrm{opt}}=\frac{\sum_{i=1}^{\ell}\Big(\big(\boldsymbol{z}_i^{\mathrm{T}}(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)^{-1}\big)\Big(\frac{\partial(\boldsymbol{W}^{-1}+\boldsymbol{\Sigma}_\dagger)}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\Big)-\widetilde{T}_\ell\Big)\Big(\big(\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{P}^{-1}\big)\Big(\frac{\partial\boldsymbol{P}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\Big)-\mathrm{Tr}\Big(\boldsymbol{P}^{-1}\frac{\partial\boldsymbol{P}}{\partial\theta_k}\Big)\Big)}{\sum_{i=1}^{\ell}\Big[\big(\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{P}^{-1}\big)\Big(\frac{\partial\boldsymbol{P}}{\partial\theta_k}\boldsymbol{P}^{-1}\boldsymbol{z}_i\Big)-\mathrm{Tr}\Big(\boldsymbol{P}^{-1}\frac{\partial\boldsymbol{P}}{\partial\theta_k}\Big)\Big]^2}.$$

Appendix E. Additional information on the preconditioners

E.1 VIF with diagonal update (VIFDU) preconditioner

The "**VIF** with **d**iagonal **u**pdate" (VIFDU) preconditioner is given by

$$\widehat{\boldsymbol{P}} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B} + \boldsymbol{\Sigma}_{\dagger}^{-1} = \boldsymbol{B}^{\mathrm{T}}\big(\boldsymbol{W} + \boldsymbol{D}^{-1} - \boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\big)\boldsymbol{B}.$$

Linear solves with $\widehat{\boldsymbol{P}}$

$$\widehat{\boldsymbol{P}}^{-1}\boldsymbol{w} = \boldsymbol{B}^{-1}\big((\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1} + (\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\big)\boldsymbol{B}^{-\mathrm{T}}\boldsymbol{w},$$

where $\boldsymbol{w} \in \mathbb{R}^n$ and $\boldsymbol{M}_3 = \boldsymbol{M} - \boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}$, are computed in $\mathcal{O}\big(n\cdot(m_v+m)\big)$ when $\boldsymbol{M}_3$ is precomputed in $\mathcal{O}\big(n\cdot(m_v\cdot m + m^2)\big)$. We can compute the log-determinant and its derivatives using Sylvester's determinant theorem

$$\begin{aligned}\log\det\big(\widehat{\boldsymbol{P}}\big) &= \log\det\Big(\boldsymbol{B}^{\mathrm{T}}\big(\boldsymbol{W}+\boldsymbol{D}^{-1}-\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\big)\boldsymbol{B}\Big)\\ &= \log\det\big(\boldsymbol{W}+\boldsymbol{D}^{-1}-\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\big)\\ &= \log\det\big(\boldsymbol{W}+\boldsymbol{D}^{-1}\big) - \log\det\big(\boldsymbol{M}\big) + \log\det\big(\boldsymbol{M}_3\big)\end{aligned}$$

$$\begin{aligned}\mathrm{Tr}\Big(\widehat{\boldsymbol{P}}^{-1}\frac{\partial\widehat{\boldsymbol{P}}}{\partial b^*(\boldsymbol{s}_i)}\Big) &= \mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\frac{\partial\boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}\Big)\\ &\quad+\mathrm{Tr}\Big(\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\frac{\partial\boldsymbol{W}}{\partial b^*(\boldsymbol{s}_i)}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\Big)\end{aligned}$$

$$\mathrm{Tr}\Big(\widehat{\boldsymbol{P}}^{-1}\frac{\partial\widehat{\boldsymbol{P}}}{\partial\theta_k}\Big) = \mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\frac{\partial(\boldsymbol{W}+\boldsymbol{D}^{-1})}{\partial\theta_k}\Big) - \mathrm{Tr}\Big(\boldsymbol{M}^{-1}\frac{\partial\boldsymbol{M}}{\partial\theta_k}\Big) + \mathrm{Tr}\Big(\boldsymbol{M}_3^{-1}\frac{\partial\boldsymbol{M}_3}{\partial\theta_k}\Big)$$

$$\begin{aligned}\mathrm{Tr}\Big(\widehat{\boldsymbol{P}}^{-1}\frac{\partial\widehat{\boldsymbol{P}}}{\partial\xi_l}\Big) &= \mathrm{Tr}\Big((\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\frac{\partial\boldsymbol{W}}{\partial\xi_l}\Big)\\ &\quad+\mathrm{Tr}\Big(\boldsymbol{M}_3^{-1}\boldsymbol{\Sigma}_{mn}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\frac{\partial\boldsymbol{W}}{\partial\xi_l}(\boldsymbol{W}+\boldsymbol{D}^{-1})^{-1}\boldsymbol{D}^{-1}\boldsymbol{B}\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\Big)\end{aligned}$$

in $\mathcal{O}\big(n\cdot(m_v\cdot m+m^2)\big)$.

E.2 FITC preconditioner

The **FITC** preconditioner is given by

$$\widehat{\boldsymbol{P}} = \mathrm{diag}(\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\Sigma}_{kn}) + \boldsymbol{W}^{-1} + \boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\Sigma}_{kn} = \boldsymbol{D}_V + \boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\Sigma}_{kn},$$

where $\boldsymbol{D}_V = \mathrm{diag}(\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\Sigma}_{kn}) + \boldsymbol{W}^{-1} \in \mathbb{R}^{n\times n}$, $\boldsymbol{\Sigma}_k = \big[c_{\boldsymbol{\theta}}(\hat{\boldsymbol{s}}_i,\hat{\boldsymbol{s}}_j)\big]_{i=1:k,j=1:k} \in \mathbb{R}^{k\times k}$ is a covariance matrix and $\boldsymbol{\Sigma}_{kn} = \big[c_{\boldsymbol{\theta}}(\hat{\boldsymbol{s}}_i,\boldsymbol{s}_j)\big]_{i=1:k,j=1:n} \in \mathbb{R}^{k\times n}$ a cross-covariance matrix with respect to the set of inducing points $\widehat{\mathcal{S}} = \{\hat{\boldsymbol{s}}_1,\dots,\hat{\boldsymbol{s}}_k\}$. This formulation allows for flexibility in selecting inducing points, enabling the use of a different or larger set of inducing points than those employed in the VIF. Such flexibility can lead to a more effective preconditioner compared to relying solely on the inducing points defined by the VIF structure. The construction of the FITC preconditioner requires $\mathcal{O}(n\cdot k^2)$ operations and linear solves $\widehat{\boldsymbol{P}}^{-1}\boldsymbol{w} = \boldsymbol{D}_V^{-1}\boldsymbol{w} - \boldsymbol{D}_V^{-1}\boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{M}_V^{-1}\boldsymbol{\Sigma}_{kn}\boldsymbol{D}_V^{-1}\boldsymbol{w}$, where $\boldsymbol{w}\in\mathbb{R}^n$ and $\boldsymbol{M}_V = \boldsymbol{\Sigma}_k + \boldsymbol{\Sigma}_{kn}\boldsymbol{D}_V^{-1}\boldsymbol{\Sigma}_{kn}^{\mathrm{T}}$, are computed in $\mathcal{O}(n\cdot k)$ when $\boldsymbol{M}_V$ is precomputed in $\mathcal{O}(n\cdot k^2)$. We can compute the

log-determinant and its derivatives using Sylvester's determinant theorem

$$\log\det\left(\widehat{\boldsymbol{P}}\right) = \log\det\left(\boldsymbol{D}_V\right) - \log\det\left(\boldsymbol{\Sigma}_k\right) + \log\det\left(\boldsymbol{M}_V\right)$$

$$\mathrm{Tr}\left(\widehat{\boldsymbol{P}}^{-1}\frac{\partial\widehat{\boldsymbol{P}}}{\partial b^*(\boldsymbol{s}_i)}\right) = \mathrm{Tr}\left(\boldsymbol{D}_V^{-1}\frac{\partial\boldsymbol{W}^{-1}}{\partial b^*(\boldsymbol{s}_i)}\right) + \mathrm{Tr}\left(\boldsymbol{M}_V^{-1}\boldsymbol{\Sigma}_{kn}\boldsymbol{D}_V^{-1}\frac{\partial\boldsymbol{W}^{-1}}{\partial b^*(\boldsymbol{s}_i)}\boldsymbol{D}_V^{-1}\boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\right)$$

$$\mathrm{Tr}\left(\widehat{\boldsymbol{P}}^{-1}\frac{\partial\widehat{\boldsymbol{P}}}{\partial\theta_k}\right) = \mathrm{Tr}\left(\boldsymbol{D}_V^{-1}\frac{\partial\boldsymbol{D}_V}{\partial\theta_k}\right) - \mathrm{Tr}\left(\boldsymbol{\Sigma}_k^{-1}\frac{\partial\boldsymbol{\Sigma}_k}{\partial\theta_k}\right) + \mathrm{Tr}\left(\boldsymbol{M}_V^{-1}\frac{\partial\boldsymbol{M}_V}{\partial\theta_k}\right)$$

$$\mathrm{Tr}\left(\widehat{\boldsymbol{P}}^{-1}\frac{\partial\widehat{\boldsymbol{P}}}{\partial\xi_l}\right) = \mathrm{Tr}\left(\boldsymbol{D}_V^{-1}\frac{\partial\boldsymbol{W}^{-1}}{\partial\xi_l}\right) + \mathrm{Tr}\left(\boldsymbol{M}_V^{-1}\boldsymbol{\Sigma}_{kn}\boldsymbol{D}_V^{-1}\frac{\partial\boldsymbol{W}^{-1}}{\partial\xi_l}\boldsymbol{D}_V^{-1}\boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\right)$$

in $\mathcal{O}(n\cdot k^2)$. Sampling from a normal distribution with mean $\mathbf{0}$ and covariance $\widehat{\boldsymbol{P}}$ can be done by using the reparameterization trick described in Gardner et al. (2018a, Appendix C.1). If $\boldsymbol{\epsilon}_1$ is a standard normal vector in $\mathbb{R}^k$ and $\boldsymbol{\epsilon}_2$ is a standard normal vector in $\mathbb{R}^n$, then the expression $\left(\boldsymbol{D}_V^{\frac{1}{2}}\boldsymbol{\epsilon}_2 + \boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-\frac{1}{2}}\boldsymbol{\epsilon}_1\right)$ is a sample from $\mathcal{N}(\mathbf{0}, \boldsymbol{D}_V + \boldsymbol{\Sigma}_{kn}^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\Sigma}_{kn})$. The computational complexity of this procedure is $\mathcal{O}(n\cdot k)$.

Appendix F. Convergence analysis of the preconditioned CG method

Lemma 7 ***Error bound for Vecchia approximation of*** $\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}$

Under Assumptions 1–3, let $(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}$ *be a Vecchia approximation with* m_v *neighbors for* $\boldsymbol{\Sigma}-\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}$*, where* $\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}$ *is the rank-m approximation of a covariance matrix* $\boldsymbol{\Sigma}$ *with eigenvalues* $\lambda_1 \geq ... \geq \lambda_n > 0$ *as described in Section 2.1. Then, the following inequality holds*

$$||(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}||_2 \leq (\alpha\cdot\lambda_{m+1}\cdot m_v)^n,$$

where $\alpha > 1$ *is a constant.*

Proof

Since $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}$ is symmetric positive definite, its singular values coincide with its eigenvalues. Therefore, for the spectral norm,

$$\left\|(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}\right\|_2 = \frac{1}{\lambda_{\min}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})} = \frac{1}{\sigma_{\min}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})},$$

where $\lambda_{\min}$ and $\sigma_{\min}$ denote the smallest eigenvalue and smallest singular value, respectively.

Furthermore, we apply the lower bound for the smallest singular value of a matrix $\mathbf{A}\in\mathbb{R}^{n\times n}$ as established by Shun (2021); Yi-Sheng and Dun-He (1997)

$$\sigma_{\min}(\mathbf{A}) \geq |\det\mathbf{A}|\cdot\left(\frac{n-1}{\|\mathbf{A}\|_F^2}\right)^{(n-1)/2},$$

where $\|\cdot\|_F$ is the Frobenius-norm. Applying this bound, we obtain

$$\begin{aligned}\frac{1}{\sigma_{\min}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})} &\leq |\det(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}|\cdot\left(\frac{\|\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\|_F^2}{n-1}\right)^{(n-1)/2}\\ &= |\det(\boldsymbol{D})|\cdot\left(\frac{\|\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\|_F^2}{n-1}\right)^{(n-1)/2}.\end{aligned}$$

By Katzfuss and Guinness (2021), we know that, for some constant $\alpha_0 > 1$, the matrix $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}$ contains at most $\alpha_0 \cdot n \cdot m_v^2$ non-zero elements and therefore, we obtain

$$\begin{aligned}|\det(\boldsymbol{D})| \cdot \left(\frac{\|\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\|_F^2}{n-1}\right)^{(n-1)/2} &\leq |\det(\boldsymbol{D})| \cdot \left(\frac{\alpha_1 \cdot n \cdot m_v^2}{n-1}\right)^{(n-1)/2} \\ &\leq \alpha_1^n \cdot |\det(\boldsymbol{D})| \cdot m_v^{n-1} \cdot \Big(1 + \frac{1}{n-1}\Big)^{(n-1)/2} \\ &\leq \alpha_1^n \cdot |\det(\boldsymbol{D})| \cdot m_v^{n-1} \cdot e^1 = \alpha_2^n \cdot |\det(\boldsymbol{D})| \cdot m_v^n,\end{aligned}$$

where $\alpha_1 = \alpha_0 \cdot (\max_{i,j=1,\dots,n} \left[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B}\right]_{i,j})^2$ and $\alpha_2 = \alpha_1 \cdot e^{1/n}$. Furthermore, we have by Sun et al. (2015)

$$\begin{aligned}|\det(\boldsymbol{D})| &\leq (\max_{i=1,\dots,n} |D_i|)^n \leq (\max_{i=1,\dots,n} |\boldsymbol{\Sigma}_{ii} - \boldsymbol{\Sigma}_{mi}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mi}|)^n \\ &\leq \big(\lambda_{\max}(\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn})\big)^n \leq \alpha_3^n \cdot \lambda_{\max}(\boldsymbol{\Sigma} - \widehat{\boldsymbol{\Sigma}}^m)^n \leq \alpha_3^n \cdot \lambda_{m+1}^n,\end{aligned}$$

where $\alpha_3 > 1$ is a constant, $\widehat{\boldsymbol{\Sigma}}^m \in \mathbb{R}^{n\times n}$ is the best rank-m approximation of $\boldsymbol{\Sigma}$ given by the m-truncated eigenvalue decomposition (see the Eckart-Young-Mirsky theorem for the spectral norm (Eckart and Young, 1936)), and $\lambda_{\max}$ the largest eigenvalue. Hence, we obtain

$$||(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}||_2 \leq \alpha_2^n \cdot |\det(\boldsymbol{D})| \cdot m_v^n \leq (\alpha \cdot \lambda_{m+1} \cdot m_v)^n,$$

where $\alpha = \alpha_2 \cdot \alpha_3$. ■

Lemma 8 ***Bound for the Condition Number of*** $\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})$

Let $\boldsymbol{\Sigma}_\dagger \in \mathbb{R}^{n\times n}$ be a VIF approximation with m inducing points and m_v Vecchia neighbors of a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$, where $\boldsymbol{\Sigma}$ has eigenvalues $\lambda_1 \geq ... \geq \lambda_n > 0$. Furthermore, let $\widehat{\boldsymbol{P}} \in \mathbb{R}^{n\times n}$ be the VIFDU preconditioner for $\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W}$. Under Assumptions 1–3, the condition number $\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\big)$ is bounded by

$$\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\big) \leq \Big(1 + \alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1} \cdot m_v)^n\big) \cdot \big(\sqrt{n} \cdot m_v + \max(\boldsymbol{W})\big)\Big)^2,$$

where $\alpha > 1$ is a constant.

Proof We use the identity

$$\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\big) = ||\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})||_2 \cdot ||(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\widehat{\boldsymbol{P}}||_2,$$

For $\boldsymbol{E} = \widehat{\boldsymbol{P}} - (\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})$, we obtain

$$\begin{aligned}\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\big) &= ||\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})||_2 \cdot ||(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\widehat{\boldsymbol{P}}||_2 \\ &= ||\widehat{\boldsymbol{P}}^{-1}(\widehat{\boldsymbol{P}} - \boldsymbol{E})||_2 \cdot ||(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\big((\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W}) + \boldsymbol{E}\big)||_2 \\ &= ||\boldsymbol{I}_n - \widehat{\boldsymbol{P}}^{-1}\boldsymbol{E}||_2 \cdot ||\boldsymbol{I}_n + (\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}\boldsymbol{E}||_2.\end{aligned}$$

Applying Cauchy-Schwarz and the triangle inequality, we have

$$\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\big) \leq \big(1 + ||\widehat{\boldsymbol{P}}^{-1}||_2 \cdot ||\boldsymbol{E}||_2\big) \cdot \big(1 + ||(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}||_2 \cdot ||\boldsymbol{E}||_2\big).$$

Furthermore, we obtain

$$||(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})^{-1}||_2 = \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger^{-1}) + \min(\boldsymbol{W})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger^{-1})} = ||\boldsymbol{\Sigma}_\dagger||_2$$

and

$$||\widehat{\boldsymbol{P}}^{-1}||_2 = \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger^{-1}) + \sigma_{\min}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger^{-1})} = ||\boldsymbol{\Sigma}_\dagger||_2.$$

Therefore, we obtain

$$\begin{aligned}
\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1} + \boldsymbol{W})\big) &\leq \big(1 + ||\boldsymbol{\Sigma}_\dagger||_2 \cdot ||\boldsymbol{E}||_2\big)^2 = \big(1 + ||\boldsymbol{\Sigma}_\dagger||_2 \cdot ||\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B} - \boldsymbol{W}||_2\big)^2 \\
&\leq \big(1 + ||\boldsymbol{\Sigma}_\dagger||_2 \cdot (||\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B}||_2 + \max(\boldsymbol{W}))\big)^2 \\
&\leq \big(1 + ||\boldsymbol{\Sigma}_\dagger||_2 \cdot (||\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B}||_F + \max(\boldsymbol{W}))\big)^2 \\
&\leq \Big(1 + ||\boldsymbol{\Sigma}_\dagger||_2 \cdot (\alpha_1 \cdot \sqrt{n} \cdot m_v + \max(\boldsymbol{W}))\Big)^2 \\
&= \Big(1 + ||\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} + (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}||_2 \cdot (\alpha_1 \cdot \sqrt{n} \cdot m_v + \max(\boldsymbol{W}))\Big)^2 \\
&\leq \Big(1 + (||\boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}||_2 + ||(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}||_2) \\
&\quad \cdot (\alpha_1 \cdot \sqrt{n} \cdot m_v + \max(\boldsymbol{W}))\Big)^2 \\
&\leq \Big(1 + \Big(||\boldsymbol{\Sigma}||_2 + (\alpha_2 \cdot \lambda_{m+1} \cdot m_v)^n\Big) \cdot \Big(\alpha_1 \cdot \sqrt{n} \cdot m_v + \max(\boldsymbol{W})\Big)\Big)^2 \\
&= \Big(1 + \Big(\lambda_1 + (\alpha_2 \cdot \lambda_{m+1} \cdot m_v)^n\Big) \cdot \Big(\alpha_1 \cdot \sqrt{n} \cdot m_v + \max(\boldsymbol{W})\Big)\Big)^2 \\
&\leq \Big(1 + \alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1} \cdot m_v)^n\big) \cdot \big(\sqrt{n} \cdot m_v + \max(\boldsymbol{W})\big)\Big)^2,
\end{aligned}$$

where, in the fourth inequality, we again invoke the result of Katzfuss and Guinness (2021), which guarantees the existence of a constant $\alpha_0 > 1$ such that the matrix $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}^{-1}\boldsymbol{B}$ contains at most $\alpha_0 \cdot n \cdot m_v^2$ non-zero entries. This implies $\alpha_1 = \sqrt{\alpha_0}\, \max_{i,j=1,\dots,n} \big|[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{B}]_{i,j}\big|$. Furthermore, in the sixth inequality we apply Lemma 7 with some constant $\alpha_2 > 1$. Finally, the constant $\alpha > 1$ is chosen such that $\alpha = \max(\alpha_1^{1/n}, \alpha_2, \alpha_1^{1/n} \cdot \alpha_2)$.

■

Proof [Proof of Theorem 5] By using a well-known CG convergence result (Trefethen and Bau, 2022), which bounds the error in terms of the conditioning number, and Lemma 8, we

obtain

$$\begin{aligned}\frac{\|\mathbf{u}^* - \mathbf{u}_k\|_{\boldsymbol{\Sigma}_\dagger^{-1}+\boldsymbol{W}}}{\|\mathbf{u}^* - \mathbf{u}_0\|_{\boldsymbol{\Sigma}_\dagger^{-1}+\boldsymbol{W}}} &\leq 2 \cdot \left(\frac{\sqrt{\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1}+\boldsymbol{W})\big)}-1}{\sqrt{\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger^{-1}+\boldsymbol{W})\big)}+1}\right)^k \\ &\leq 2 \cdot \left(\frac{1+\alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1}\cdot m_v)^n\big)\cdot\big(\sqrt{n}\cdot m_v + \max(\boldsymbol{W})\big)-1}{1+\alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1}\cdot m_v)^n\big)\cdot\big(\sqrt{n}\cdot m_v + \max(\boldsymbol{W})\big)+1}\right)^k \\ &= 2 \cdot \left(\frac{\alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1}\cdot m_v)^n\big)\cdot\big(\sqrt{n}\cdot m_v + \max(\boldsymbol{W})\big)}{\alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1}\cdot m_v)^n\big)\cdot\big(\sqrt{n}\cdot m_v + \max(\boldsymbol{W})\big)+2}\right)^k \\ &\leq 2 \cdot \left(\frac{1}{1+\Big(\alpha^n \cdot \big(\lambda_1 + (\lambda_{m+1}\cdot m_v)^n\big)\cdot\big(\sqrt{n}\cdot m_v + \max(\boldsymbol{W})\big)\Big)^{-1}}\right)^k.\end{aligned}$$

■

Lemma 9 ***Bound for the Condition Number of*** $\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})$

Let $\boldsymbol{\Sigma}_\dagger \in \mathbb{R}^{n\times n}$ *be a VIF approximation with* m *inducing points and* m_v *Vecchia neighbors of a covariance matrix* $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$*, where* $\boldsymbol{\Sigma}$ *has eigenvalues* $\lambda_1 \geq ... \geq \lambda_n > 0$*. Furthermore, let* $\widehat{\boldsymbol{P}} \in \mathbb{R}^{n\times n}$ *be the FITC preconditioner for* $\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1}$ *with the same set of inducing points as those used for* $\boldsymbol{\Sigma}_\dagger$*. Under Assumptions 1–3, the condition number* $\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big)$ *is bounded by*

$$\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big) \leq \Big(1 + \max(\boldsymbol{W})\cdot(\alpha\cdot\lambda_{m+1}\cdot m_v)^n\Big)^2,$$

where $\alpha > 1$ *is a constant.*

Proof For $\boldsymbol{E} = (\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1}) - \widehat{\boldsymbol{P}}$, we obtain

$$\begin{aligned}\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big) &= ||\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})||_2 \cdot ||(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}\widehat{\boldsymbol{P}}||_2 \\ &= ||\widehat{\boldsymbol{P}}^{-1}(\widehat{\boldsymbol{P}} + \boldsymbol{E})||_2 \cdot ||(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}\big((\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1}) - \boldsymbol{E}\big)||_2 \\ &= ||\boldsymbol{I}_n + \widehat{\boldsymbol{P}}^{-1}\boldsymbol{E}||_2 \cdot ||\boldsymbol{I}_n - (\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}\boldsymbol{E}||_2.\end{aligned}$$

Applying Cauchy-Schwarz and the triangle inequality, we have

$$\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big) \leq \big(1 + ||\widehat{\boldsymbol{P}}^{-1}||_2 \cdot ||\boldsymbol{E}||_2\big)\cdot\big(1 + ||(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}||_2 \cdot ||\boldsymbol{E}||_2\big).$$

Furthermore, we obtain

$$||(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})^{-1}||_2 = \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{\Sigma}_\dagger) + \min(\boldsymbol{W}^{-1})} \leq \frac{1}{\min(\boldsymbol{W}^{-1})} = \max(\boldsymbol{W})$$

and

$$\begin{aligned}||\widehat{\boldsymbol{P}}^{-1}||_2 &= \frac{1}{\sigma_{\min}(\boldsymbol{D}_I + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn} + \boldsymbol{W}^{-1})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{D}_I + \boldsymbol{\Sigma}_{mn}^{\mathrm{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn}) + \min(\boldsymbol{W}^{-1})} \\ &\leq \frac{1}{\min(\boldsymbol{W}^{-1})} = \max(\boldsymbol{W}),\end{aligned}$$

where $\boldsymbol{D}_I = \text{diag}(\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{mn}^{\text{T}}\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\Sigma}_{mn})$. Therefore, we obtain

$$\begin{aligned}\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big) &\leq \big(1 + \max(\boldsymbol{W}) \cdot ||\boldsymbol{E}||_2\big)^2 = \big(1 + \max(\boldsymbol{W}) \cdot ||(\boldsymbol{B}^{\text{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1} - \boldsymbol{D}_I||_2\big)^2 \\ &\leq \big(1 + \max(\boldsymbol{W}) \cdot ||(\boldsymbol{B}^{\text{T}}\boldsymbol{D}^{-1}\boldsymbol{B})^{-1}||_2\big)^2 \\ &\leq \Big(1 + \max(\boldsymbol{W}) \cdot (\alpha \cdot \lambda_{m+1} \cdot m_v)^n\Big)^2,\end{aligned}$$

where we used Lemma 7 with a constant $\alpha > 1$. ■

Proof [Proof of Theorem 6] By proceeding analogue to the proof of Theorem 5 and by using Lemma 9, we obtain

$$\begin{aligned}\frac{\|\mathbf{u}^* - \mathbf{u}_k\|_{\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1}}}{\|\mathbf{u}^* - \mathbf{u}_0\|_{\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1}}} &\leq 2 \cdot \left(\frac{\sqrt{\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big)} - 1}{\sqrt{\kappa\big(\widehat{\boldsymbol{P}}^{-1}(\boldsymbol{\Sigma}_\dagger + \boldsymbol{W}^{-1})\big)} + 1}\right)^k \\ &\leq 2 \cdot \left(\frac{1 + \max(\boldsymbol{W}) \cdot (\alpha \cdot \lambda_{m+1} \cdot m_v)^n - 1}{1 + \max(\boldsymbol{W}) \cdot (\alpha \cdot \lambda_{m+1} \cdot m_v)^n + 1}\right)^k \\ &= 2 \cdot \left(\frac{\max(\boldsymbol{W}) \cdot (\alpha \cdot \lambda_{m+1} \cdot m_v)^n}{\max(\boldsymbol{W}) \cdot (\alpha \cdot \lambda_{m+1} \cdot m_v)^n + 2}\right)^k \\ &\leq 2 \cdot \left(\frac{1}{1 + \max(\boldsymbol{W})^{-1} \cdot (\alpha \cdot \lambda_{m+1} \cdot m_v)^{-n}}\right)^k.\end{aligned}$$

■

7. Additional results

		Vecchia	FITC	VIF
Training	$d = 2$	4	13	24
	$d = 5$	8	38	39
	$d = 10$	16	91	86
	$d = 20$	35	272	244
	$d = 50$	129	1081	811
	$d = 100$	396	3179	2840

		Vecchia	FITC	VIF
Prediction	$d = 2$	0.6	0.7	0.9
	$d = 5$	0.7	0.8	1.2
	$d = 10$	0.7	0.9	1.4
	$d = 20$	1.0	1.3	1.6
	$d = 50$	1.3	1.8	2.0
	$d = 100$	1.6	2.2	2.4

Table 4: Average runtimes in seconds for the experiments reported in Figure 2 in Section 7.1

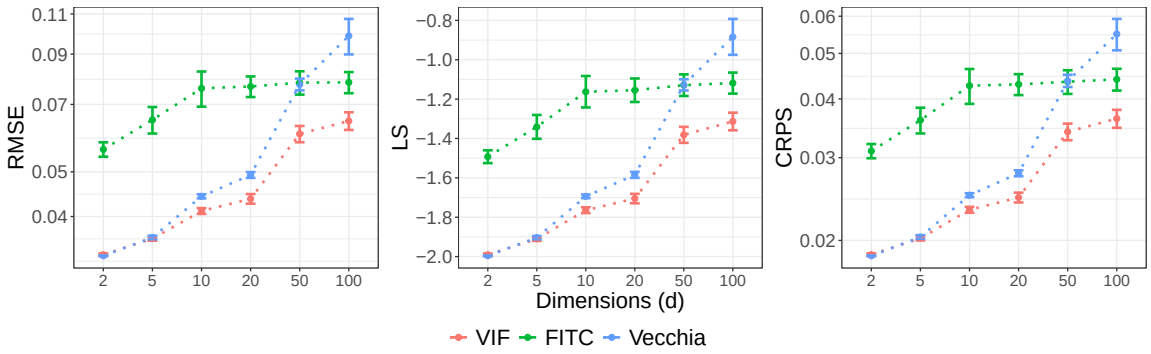

Figure 10: RMSE, log-score (LS), and CRPS (mean ± 2 standard errors) for VIF ($m_v = 30$ & $m = 200$), FITC ($m = 200$), and Vecchia ($m_v = 60$) approximation for varying dimensions d using a 3/2-Matérn kernel.

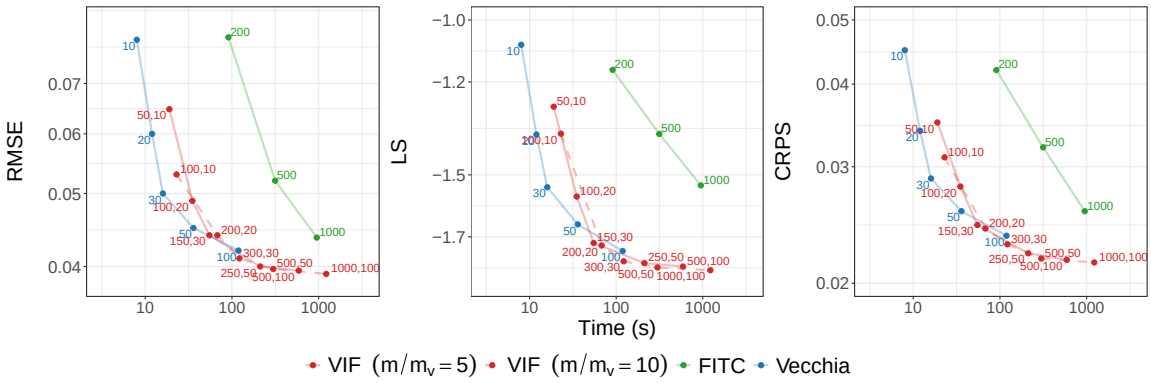

Figure 11: RMSE, log-score (LS), and CRPS (mean) versus runtime (training + prediction) in seconds (s) for VIF, FITC, and Vecchia approximations across varying numbers of inducing points $m \in \{50, 100, 150, 200, 250, 300, 500, 1000\}$ and Vecchia neighbors $m_v \in \{10, 20, 30, 50, 100\}$. For VIF, two parameter ratios are illustrated: $\frac{m}{m_v} = 5$ (solid) and $\frac{m}{m_v} = 10$ (dashed). Experiments are conducted with input dimension $d = 10$ using a Matérn-3/2 kernel, and the corresponding values of m and m_v are annotated in the figure.

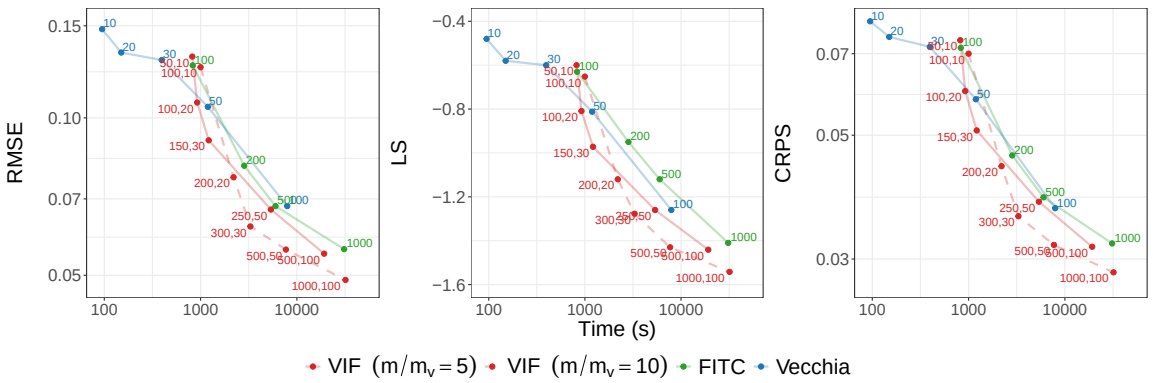

Figure 12: RMSE, log-score (LS), and CRPS (mean) versus runtime (training + prediction) in seconds (s) for VIF, FITC, and Vecchia approximations across varying numbers of inducing points $m \in \{50, 100, 150, 200, 250, 300, 500, 1000\}$ and Vecchia neighbors $m_v \in \{10, 20, 30, 50, 100\}$. For VIF, two parameter ratios are illustrated: $\frac{m}{m_v} = 5$ (solid) and $\frac{m}{m_v} = 10$ (dashed). Experiments are conducted with input dimension $d = 100$ using a Matérn-3/2 kernel, and the corresponding values of m and m_v are annotated in the figure.

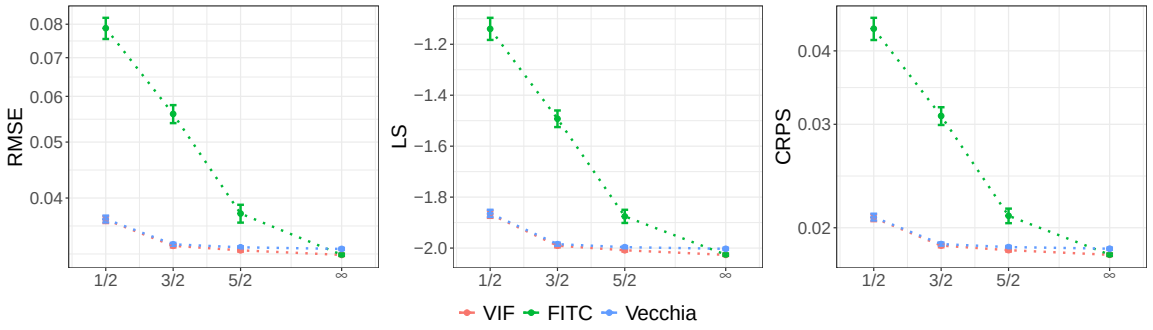

Figure 13: RMSE (log-scale), log-score (LS), and CRPS (log-scale) (mean ± 2 standard errors) for VIF ($m_v = 30$ & $m = 200$), FITC ($m = 200$), and Vecchia ($m_v = 30$) approximations for 1/2-Matérn, 3/2-Matérn, 5/2-Matérn, and Gaussian (∞-Matérn) ARD kernels when $d = 2$.

	Figures 2, 3, and 13				Figure 14
Kernel:	1/2-Matern	3/2-Matern	5/2-Matern	Gaussian	3/2-Matern
$d = 2$	$(0.07, 0.30)^{\mathrm{T}}$	$(0.10, 0.22)^{\mathrm{T}}$	$(0.12, 0.21)^{\mathrm{T}}$	$(0.13, 0.19)^{\mathrm{T}}$	$(0.20, 0.36)^{\mathrm{T}}$
$d = 5$		$(0.13, 0.473, ..., 1.5)^{\mathrm{T}}$			$(0.23, 0.41, ..., 0.96)^{\mathrm{T}}$
$d = 10$	$(0.15, 0.389, ..., 2.3)^{\mathrm{T}}$	$(0.25, 0.467, ..., 2.2)^{\mathrm{T}}$	$(0.27, 0.473, ..., 2.1)^{\mathrm{T}}$	$(0.28, 0.471, ..., 2.0)^{\mathrm{T}}$	$(0.24, 0.43, ..., 1.96)^{\mathrm{T}}$
$d = 20$		$(0.50, 0.763, ..., 5.5)^{\mathrm{T}}$			$(0.25, 0.45, ..., 4.00)^{\mathrm{T}}$
$d = 50$		$(0.55, 0.661, ..., 6.0)^{\mathrm{T}}$			$(0.25, 0.45, ..., 10.16)^{\mathrm{T}}$
$d = 100$		$(0.60, 0.665, ..., 7.0)^{\mathrm{T}}$			$(0.25, 0.46, ..., 20.45)^{\mathrm{T}}$

Table 5: Length scale parameters $\boldsymbol{\lambda}$ used in Figures 2, 3, 13, and 14 for various dimensions d and kernels ("..." means linearly interpolated).

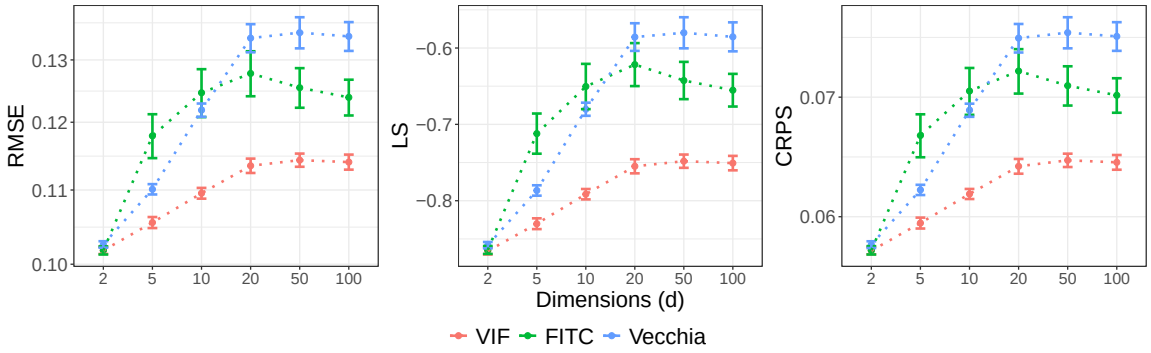

Figure 14: RMSE, log-score (LS), and CRPS (mean ± 2 standard errors) for VIF ($m_v = 30$ & $m = 200$), FITC ($m = 200$), and Vecchia ($m_v = 30$) approximations for various dimensions d using an error variance of 0.01 and length scale parameters chosen such that the covariance remains approximately equal (to the one of a Gaussian kernel with length scales $\boldsymbol{\lambda} = (0.35, 0.4, 0.45, 0.5, 0.55)^{\mathrm{T}}$) at the average distance among two randomly chosen points.

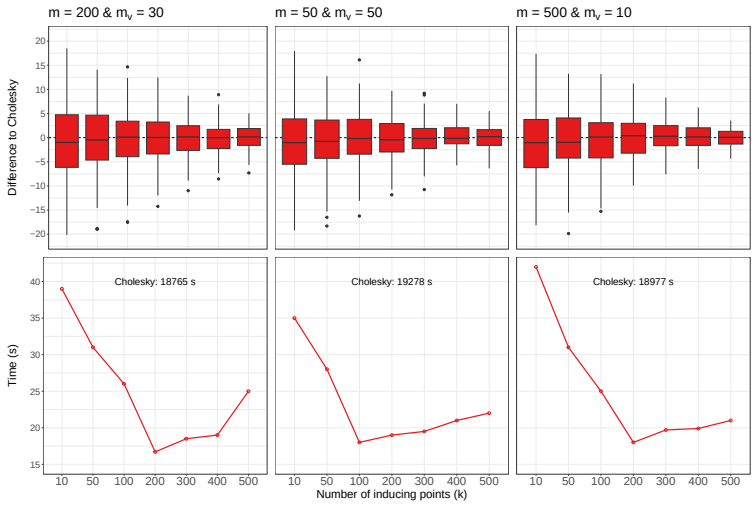

Figure 15: Log-marginal likelihood differences relative to Cholesky-based computations and runtime of the FITC preconditioner for varying numbers of inducing points k for three different VIF approximations using $\ell = 50$ sample vectors.

Preconditioner	CG convergence tolerance (δ)	$\ell = 10$	$\ell = 50$	$\ell = 100$
FITC	$\delta = 1$	8.813 (10 s)	3.456 (16 s)	2.626 (20 s)
	$\delta = 0.1$	8.792 (16 s)	3.410 (20 s)	2.610 (23 s)
	$\delta = 0.01$	8.723 (19 s)	3.389 (23 s)	2.602 (26 s)
	$\delta = 0.001$	8.723 (33 s)	3.389 (38 s)	2.600 (41 s)
	$\delta = 0.0001$	8.723 (40 s)	3.389 (46 s)	2.600 (49 s)
VIFDU	$\delta = 1$	12.295 (22 s)	5.301 (40 s)	4.887 (72 s)
	$\delta = 0.1$	12.269 (24 s)	5.263 (45 s)	4.849 (76 s)
	$\delta = 0.01$	12.234 (27 s)	5.218 (48 s)	4.804 (80 s)
	$\delta = 0.001$	12.234 (31 s)	5.217 (52 s)	4.803 (85 s)
	$\delta = 0.0001$	12.233 (35 s)	5.217 (56 s)	4.803 (89 s)

Table 6: Accuracy-runtime comparison of preconditioners: RMSE between log-marginal likelihoods computed using iterative methods and those computed using a Cholesky decomposition and runtime for the VIFDU and FITC preconditioners and varying numbers of sample vectors $\ell \in \{10, 50, 100\}$ and CG convergence tolerances $\delta \in \{0.0001, 0.001, 0.01, 0.1, 1\}$. We used a binary likelihood, $n = 100{,}000$ samples, and a VIF approximation with $m = 200$ & $m_v = 30$.

Data	Accuracy measure	**VIF** $m_v = 30$ $m = 200$	
3dRoad $n = 434{,}874$ $d = 3$	RMSE	**0.145 ± 0.002**	**0.143 ± 0.002**
	CRPS	**0.068 ± 0.002**	**0.067 ± 0.002**
	LS	-0.625 ± 0.010	**-0.746 ± 0.006**
	ν	1.5	0.696 ± 0.004
	Time	376 s	865 s
KEGGU $n = 63{,}608$ $d = 26$	RMSE	**0.094 ± 0.008**	**0.097 ± 0.002**
	CRPS	**0.030 ± 0.002**	**0.028 ± 0.002**
	LS	**-1.081 ± 0.072**	**-1.113 ± 0.056**
	ν	1.5	2.137 ± 0.950
	Time	738 s	1214 s
KEGG $n = 48{,}827$ $d = 18$	RMSE	**0.100 ± 0.010**	**0.104 ± 0.012**
	CRPS	**0.041 ± 0.004**	**0.040 ± 0.002**
	LS	**-1.082 ± 0.070**	**-1.106 ± 0.034**
	ν	1.5	2.769 ± 0.622
	Time	383 s	687 s
Elevators $n = 16{,}599$ $d = 17$	RMSE	**0.355 ± 0.004**	**0.351 ± 0.004**
	CRPS	**0.196 ± 0.004**	**0.194 ± 0.002**
	LS	**0.388 ± 0.012**	**0.389 ± 0.006**
	ν	1.5	0.912 ± 0.218
	Time	711 s	935 s
Protein $n = 45{,}730$ $d = 8$	RMSE	0.516 ± 0.006	**0.499 ± 0.006**
	CRPS	0.259 ± 0.002	**0.250 ± 0.002**
	LS	0.667 ± 0.016	**0.627 ± 0.014**
	ν	1.5	0.567 ± 0.022
	Time	547 s	835 s
Ailerons $n = 13{,}750$ $d = 33$	RMSE	0.399 ± 0.019	**0.377 ± 0.007**
	CRPS	0.208 ± 0.008	**0.200 ± 0.003**
	LS	0.436 ± 0.030	**0.375 ± 0.014**
	ν	1.5	0.674 ± 0.033
	Time	528 s	602 s

Data	Accuracy measure	**VIF** $m_v = 30$ $m = 200$	
Kin40K $n = 40{,}000$ $d = 8$	RMSE	0.114 ± 0.002	**0.111 ± 0.002**
	CRPS	0.059 ± 0.002	**0.055 ± 0.002**
	LS	-0.808 ± 0.012	**-0.885 ± 0.024**
	ν	1.5	7.426 ± 0.842
	Time	333 s	1009 s
Bike $n = 17{,}379$ $d = 12$ (Poisson)	RMSE	**38.904 ± 1.332**	**39.065 ± 1.346**
	CRPS	**17.162 ± 0.454**	**17.477 ± 0.381**
	LS	**4.676 ± 0.030**	**4.672 ± 0.030**
	ν	1.5	0.959 ± 0.026
	Time	1534 s	2017 s
House $n = 20{,}640$ $d = 8$ (Student-t)	RMSE	0.214 ± 0.008	**0.203 ± 0.006**
	CRPS	0.103 ± 0.002	**0.099 ± 0.003**
	LS	0.092 ± 0.076	**-0.184 ± 0.009**
	ν	1.5	0.469 ± 0.023
	Time	1932 s	3021 s
Power $n = 52{,}417$ $d = 5$ (Gamma)	RMSE	0.218 ± 0.002	**0.201 ± 0.002**
	CRPS	0.121 ± 0.001	**0.111 ± 0.001**
	LS	-0.084 ± 0.010	**-0.140 ± 0.008**
	ν	1.5	0.240 ± 0.018
	Time	5035 s	4452 s
Water-Vapor $n = 100{,}000$ $d = 2$ (Gamma)	RMSE	**0.102 ± 0.001**	**0.101 ± 0.001**
	CRPS	**0.056 ± 0.001**	**0.055 ± 0.001**
	LS	**-0.728 ± 0.022**	-0.699 ± 0.020
	ν	1.5	0.383 ± 0.005
	Time	3975 s	5084 s

Table 7: RMSE, CRPS, log-score (LS), ν (mean ± 2 standard errors) and runtime in seconds (s) for a fixed 3/2-Matérn kernel (left columns) and Matérn kernel with the smoothness parameter estimated (right columns). Bold indicates the best mean; other means are in bold if within two standard errors.

Data	Performance Measure	**VIF** $m_v = 30$ $m = 200$	**Vecchia** $m_v = 30$	**FITC** $m = 200$
3dRoad $n = 434{,}874$ $d = 3$	RMSE	**0.145 ± 0.002**	**0.145 ± 0.001**	0.554 ± 0.006
	CRPS	**0.068 ± 0.002**	**0.068 ± 0.002**	0.300 ± 0.003
	LS	**-0.625 ± 0.010**	**-0.625 ± 0.010**	0.831 ± 0.011
	Time	376 s	45 s	144 s
KEGGU $n = 63{,}608$ $d = 26$	RMSE	**0.094 ± 0.008**	**0.093 ± 0.006**	0.120 ± 0.023
	CRPS	**0.030 ± 0.002**	**0.030 ± 0.001**	0.046 ± 0.019
	LS	**-1.081 ± 0.072**	**-1.060 ± 0.063**	-1.052 ± 0.123
	Time	738 s	233 s	628 s
KEGG $n = 48{,}827$ $d = 18$	RMSE	**0.100 ± 0.010**	**0.102 ± 0.005**	NA convergence issues
	CRPS	**0.041 ± 0.004**	**0.042 ± 0.001**	
	LS	**-1.082 ± 0.070**	**-1.050 ± 0.021**	
	Time	383 s	60 s	
Elevators $n = 16{,}599$ $d = 17$	RMSE	**0.355 ± 0.004**	0.382 ± 0.009	0.453 ± 0.011
	CRPS	**0.196 ± 0.004**	0.215 ± 0.004	0.248 ± 0.012
	LS	**0.388 ± 0.012**	0.431 ± 0.017	0.542 ± 0.028
	Time	711 s	37 s	347 s
Protein $n = 45{,}730$ $d = 8$	RMSE	**0.516 ± 0.006**	**0.517 ± 0.006**	0.720 ± 0.004
	CRPS	**0.259 ± 0.002**	**0.259 ± 0.003**	0.402 ± 0.002
	LS	**0.667 ± 0.016**	**0.670 ± 0.017**	1.085 ± 0.005
	Time	547 s	64 s	392 s
Kin40K $n = 40{,}000$ $d = 8$	RMSE	**0.114 ± 0.002**	0.140 ± 0.002	0.422 ± 0.004
	CRPS	**0.059 ± 0.002**	0.076 ± 0.001	0.232 ± 0.002
	LS	**-0.808 ± 0.012**	-0.552 ± 0.011	0.523 ± 0.010
	Time	333 s	153 s	852 s
Ailerons $n = 13{,}750$ $d = 33$	RMSE	**0.399 ± 0.019**	0.429 ± 0.016	0.879 ± 0.240
	CRPS	**0.208 ± 0.008**	0.218 ± 0.004	0.478 ± 0.131
	LS	**0.436 ± 0.030**	0.472 ± 0.026	1.228 ± 0.381
	Time	528 s	108 s	259 s
Bike $n = 17{,}379$ $d = 12$ (Poisson)	RMSE	**38.904 ± 1.332**	**38.768 ± 1.096**	NA convergence issues
	CRPS	**17.162 ± 0.454**	**17.073 ± 0.390**	
	LS	**4.676 ± 0.030**	**4.686 ± 0.029**	
	Time	1534 s	1296 s	
House $n = 20{,}640$ $d = 8$ (Student-t)	RMSE	**0.214 ± 0.008**	**0.213 ± 0.007**	0.292 ± 0.014
	CRPS	**0.103 ± 0.002**	**0.102 ± 0.002**	0.159 ± 0.008
	LS	0.092 ± 0.076	**0.081 ± 0.074**	0.395 ± 0.124
	Time	1932 s	2205 s	1052 s
Power $n = 52{,}417$ $d = 5$ (Gamma)	RMSE	**0.218 ± 0.002**	**0.218 ± 0.003**	NA convergence issues
	CRPS	**0.121 ± 0.001**	**0.122 ± 0.002**	
	LS	**-0.084 ± 0.010**	**-0.086 ± 0.009**	
	Time	5035 s	4806 s	
WaterVapor $n = 100{,}000$ $d = 2$ (Gamma)	RMSE	**0.102 ± 0.001**	**0.102 ± 0.001**	0.636 ± 0.011
	CRPS	**0.056 ± 0.001**	**0.056 ± 0.001**	0.525 ± 0.009
	LS	**-0.728 ± 0.022**	**-0.728 ± 0.021**	0.123 ± 0.007
	Time	3975 s	2696 s	716 s

Table 8: RMSE, CRPS, log-score (LS) (mean ± 2 standard errors), and runtime in seconds (s) for the regression data sets. Bold indicates the best mean; other means are in bold if within two standard errors.

Data	Performance Measure	**VIF** $m_v = 30$ $m = 200$	**Vecchia** $m_v = 30$	**FITC** $m = 200$
Bank $n = 45{,}211$ $d = 16$	AUC	**0.806 ± 0.008**	**0.800 ± 0.007**	0.724 ± 0.010
	RMSE	**0.284 ± 0.004**	**0.282 ± 0.005**	0.340 ± 0.009
	ACC	**0.895 ± 0.002**	**0.898 ± 0.003**	0.801 ± 0.009
	LS	**0.035 ± 0.016**	**0.040 ± 0.011**	0.067 ± 0.019
	Time	1643 s	798 s	692 s
Adult $n = 48{,}842$ $d = 14$	AUC	**0.880 ± 0.008**	**0.888 ± 0.009**	NA convergence issues
	RMSE	**0.336 ± 0.006**	**0.329 ± 0.007**	
	ACC	**0.838 ± 0.006**	**0.843 ± 0.006**	
	LS	**0.003 ± 0.012**	**-0.007 ± 0.012**	
	Time	2854 s	805 s	
Credit $n = 30{,}000$ $d = 22$	AUC	**0.768 ± 0.006**	**0.770 ± 0.007**	**0.772 ± 0.005**
	RMSE	**0.378 ± 0.004**	**0.378 ± 0.005**	**0.378 ± 0.005**
	ACC	**0.808 ± 0.006**	**0.807 ± 0.006**	**0.807 ± 0.007**
	LS	**0.407 ± 0.012**	**0.402 ± 0.013**	**0.400 ± 0.014**
	Time	1161 s	518 s	378 s
MAGIC $n = 19{,}020$ $d = 9$	AUC	**0.920 ± 0.004**	**0.918 ± 0.005**	0.895 ± 0.008
	RMSE	**0.316 ± 0.004**	**0.313 ± 0.005**	0.329 ± 0.008
	ACC	**0.866 ± 0.006**	**0.871 ± 0.006**	0.830 ± 0.009
	LS	**0.006 ± 0.022**	**0.012 ± 0.025**	0.092 ± 0.032
	Time	236 s	170 s	113 s

Table 9: AUC, RMSE (Brier score), ACC, LS (mean ± 2 standard errors), and runtime in seconds (s) for the binary classification data sets. Bold indicates the best mean; other means are in bold if within two standard errors.

Data	Accuracy measure	**Pure VIF-GP**	**Linear regression + VIF-GP**	**VIF-GPBoost**
3dRoad	RMSE	**0.145 ± 0.002**	**0.145 ± 0.002**	**0.145 ± 0.002**
$n = 434{,}874$	CRPS	**0.068 ± 0.002**	**0.068 ± 0.001**	**0.068 ± 0.001**
$d = 3$	LS	**-0.625 ± 0.010**	**-0.626 ± 0.010**	**-0.629 ± 0.011**
KEGGU	RMSE	**0.094 ± 0.008**	0.131 ± 0.009	**0.098 ± 0.009**
$n = 63{,}608$	CRPS	**0.030 ± 0.002**	0.041 ± 0.002	**0.032 ± 0.001**
$d = 26$	LS	**-1.081 ± 0.072**	1.357 ± 1.138	**-1.075 ± 0.068**
KEGG	RMSE	**0.100 ± 0.010**	0.118 ± 0.020	**0.099 ± 0.009**
$n = 48{,}827$	CRPS	**0.041 ± 0.004**	**0.045 ± 0.001**	**0.043 ± 0.002**
$d = 18$	LS	**-1.082 ± 0.070**	-0.979 ± 0.037	**-1.079 ± 0.068**
Elevators	RMSE	0.355 ± 0.004	0.359 ± 0.005	**0.347 ± 0.004**
$n = 16{,}599$	CRPS	**0.196 ± 0.004**	**0.199 ± 0.005**	**0.196 ± 0.003**
$d = 17$	LS	0.388 ± 0.012	0.398 ± 0.014	**0.378 ± 0.008**
Protein	RMSE	**0.516 ± 0.006**	**0.512 ± 0.007**	**0.513 ± 0.006**
$n = 45{,}730$	CRPS	**0.259 ± 0.002**	**0.256 ± 0.003**	**0.256 ± 0.003**
$d = 8$	LS	0.667 ± 0.016	**0.649 ± 0.017**	**0.653 ± 0.019**
Kin40K	RMSE	**0.114 ± 0.002**	0.115 ± 0.002	**0.112 ± 0.002**
$n = 40{,}000$	CRPS	0.059 ± 0.002	0.059 ± 0.001	**0.057 ± 0.001**
$d = 8$	LS	-0.808 ± 0.012	-0.807 ± 0.013	**-0.877 ± 0.014**
Ailerons	RMSE	**0.399 ± 0.019**	**0.401 ± 0.017**	**0.385 ± 0.016**
$n = 13{,}750$	CRPS	**0.208 ± 0.008**	**0.209 ± 0.007**	**0.205 ± 0.005**
$d = 33$	LS	**0.436 ± 0.030**	**0.438 ± 0.026**	**0.422 ± 0.022**

Table 10: RMSE, CRPS, log-score (LS) (mean ± 2 standard errors) when using non-zero prior mean functions (linear and tree-boosting). Bold indicates the best mean; other means are in bold if within two standard errors.

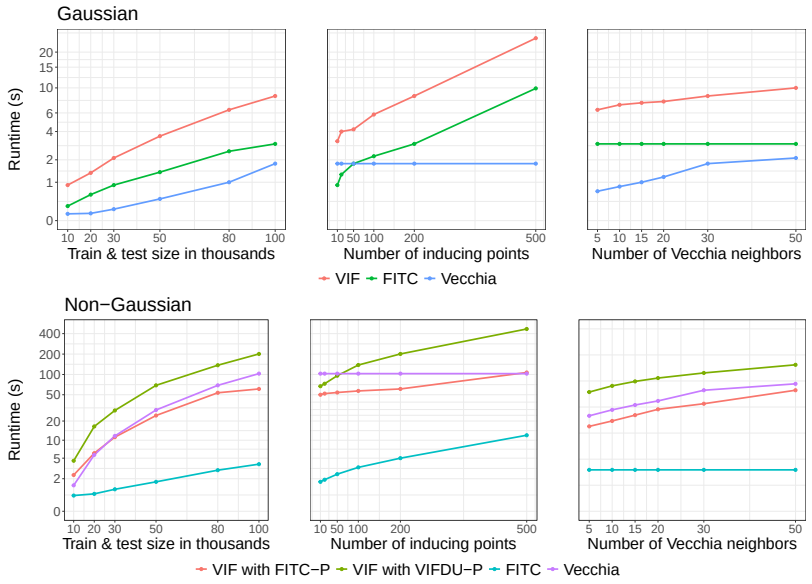

Figure 16: Time (s) for computing predictive distributions with VIF, FITC, and Vecchia approximations on simulated data for varying sample sizes n, numbers of inducing points m, and numbers of Vecchia neighbors m_v.

References

David Arthur and Sergei Vassilvitskii. K-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035, 2007.

Erlend Aune, Daniel P Simpson, and Jo Eidsvik. Parameter estimation in high dimensional Gaussian distributions. *Statistics and Computing*, 24:247–263, 2014.

Sudipto Banerjee, Alan E Gelfand, Andrew O Finley, and Huiyan Sang. Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(4):825–848, 2008.

Costas Bekas, Effrosyni Kokiopoulou, and Yousef Saad. An estimator for the diagonal of a matrix. *Applied Numerical Mathematics*, 57(11-12):1214–1229, 2007.

Alina Beygelzimer, Sham Kakade, and John Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd international conference on Machine learning*, pages 97–104, 2006.

Jian Cao, Myeongjong Kang, Felix Jimenez, Huiyan Sang, Florian Tobias Schaefer, and Matthias Katzfuss. Variational sparse inverse Cholesky approximation for latent Gaussian processes via double Kullback-Leibler minimization. In *International Conference on Machine Learning*, pages 3559–3576. PMLR, 2023.

Noel Cressie. *Statistics for spatial data*. John Wiley & Sons, 1993.

Ryan R Curtin. *Improving dual-tree algorithms.* PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2016.

Abhirup Datta, Sudipto Banerjee, Andrew O Finley, and Alan E Gelfand. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.

Timothy A Davis. *Direct methods for sparse linear systems.* SIAM, 2006.

Kun Dong, David Eriksson, Hannes Nickisch, David Bindel, and Andrew G Wilson. Scalable log determinants for Gaussian process kernel learning. *Advances in Neural Information Processing Systems*, 30, 2017.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Yury Elkin and Vitaliy Kurlin. A new near-linear time algorithm for k-nearest neighbor search using a compressed cover tree. In *International Conference on Machine Learning*, pages 9267–9311. PMLR, 2023.

Andrew O Finley, Huiyan Sang, Sudipto Banerjee, and Alan E Gelfand. Improving the performance of predictive process modeling for large datasets. *Computational Statistics & Data Analysis*, 53(8):2873–2884, 2009.

Roger Fletcher. *Practical methods of optimization.* John Wiley & Sons, 2000.

Reinhard Furrer, Marc G Genton, and Douglas Nychka. Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics*, 15(3): 502–523, 2006.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix Gaussian process inference with gpu acceleration. *Advances in Neural Information Processing Systems*, 31, 2018a.

Jacob Gardner, Geoff Pleiss, Ruihan Wu, Kilian Weinberger, and Andrew Wilson. Product kernel interpolation for scalable Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 1407–1416. PMLR, 2018b.

Joseph Guinness. Gaussian process learning via Fisher scoring of Vecchia's approximation. *Statistics and Computing*, 31(3):1–8, 2021.

Tim Gyger, Reinhard Furrer, and Fabio Sigrist. Iterative Methods for Full-Scale Gaussian Process Approximations for Large Spatial Data. *SIAM/ASA Journal on Uncertainty Quantification*, 14(1):142–167, 2026.

Helmut Harbrecht, Michael Peters, and Reinhold Schneider. On the low-rank approximation by the pivoted Cholesky decomposition. *Applied Numerical Mathematics*, 62(4):428–440, 2012.

Matthew J Heaton, Abhirup Datta, Andrew O Finley, Reinhard Furrer, Joseph Guinness, Rajarshi Guhaniyogi, Florian Gerber, Robert B Gramacy, Dorit Hammerling, Matthias Katzfuss, et al. A case study competition among methods for analyzing large spatial data. *Journal of Agricultural, Biological and Environmental Statistics*, 24:398–425, 2019.

James Hensman, Nicolò Fusi, and Neil D Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 282–290, 2013.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *Artificial intelligence and statistics*, pages 351–360. PMLR, 2015.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Myeongjong Kang and Matthias Katzfuss. Correlation-based sparse inverse Cholesky factorization for fast Gaussian-process inference. *Statistics and Computing*, 33(3):56, 2023.

Matthias Katzfuss and Joseph Guinness. A general framework for Vecchia approximations of Gaussian processes. *Statistical Science*, 36(1):124–141, 2021.

Matthias Katzfuss, Joseph Guinness, Wenlong Gong, and Daniel Zilber. Vecchia approximations of Gaussian-process predictions. *Journal of Agricultural, Biological and Environmental Statistics*, 25:383–414, 2020.

Pascal Kündig and Fabio Sigrist. Iterative methods for Vecchia-Laplace approximations for latent Gaussian process models. *Journal of the American Statistical Association*, 120 (550):1267–1280, 2025.

Cornelius Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of research of the National Bureau of Standards*, 45(4):255–282, 1950.

Christiane Lemieux. Control variates. *Wiley StatsRef: Statistics Reference Online*, pages 1–8, 2014.

Richard J Lipton, Donald J Rose, and Robert Endre Tarjan. Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16(2):346–358, 1979.

Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, 31(11):4405–4423, 2020.

Hannes Nickisch and Carl Edward Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.

Geoff Pleiss, Jacob Gardner, Kilian Weinberger, and Andrew Gordon Wilson. Constant-time predictive distributions for Gaussian processes. In *International Conference on Machine Learning*, pages 4114–4123. PMLR, 2018.

Joaquin Quinonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6: 1939–1959, 2005.

Filippo Rambelli and Fabio Sigrist. An accuracy-runtime trade-off comparison of scalable Gaussian process approximations for spatial data. *Journal of Agricultural, Biological, and Environmental Statistics (in press)*, 2026.

Carl Edward Rasmussen and Christopher K.I. Williams. *Gaussian processes for machine learning*. MIT Press Cambridge, MA, 2006.

Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

Huiyan Sang and Jianhua Z Huang. A full scale approximation of covariance functions for large spatial data sets. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 74(1):111–132, 2012.

Huiyan Sang, Mikyoung Jun, and Jianhua Z Huang. Covariance approximation for large multivariate spatial data sets with an application to multiple climate model errors. *The Annals of Applied Statistics*, pages 2519–2548, 2011.

Florian Schäfer, Matthias Katzfuss, and Houman Owhadi. Sparse Cholesky factorization by Kullback–Leibler minimization. *SIAM Journal on Scientific Computing*, 43(3):A2019–A2046, 2021a.

Florian Schäfer, Timothy John Sullivan, and Houman Owhadi. Compression, inversion, and approximate pca of dense kernel matrices at near-linear computational complexity. *Multiscale Modeling & Simulation*, 19(2):688–730, 2021b.

Xu Shun. Two new lower bounds for the smallest singular value. *arXiv preprint arXiv:2108.01221*, 2021.

Fabio Sigrist. Gaussian process boosting. *The Journal of Machine Learning Research*, 23 (1):10565–10610, 2022a.

Fabio Sigrist. Latent Gaussian model boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1894–1905, 2022b.

Giora Simchoni and Saharon Rosset. Integrating random effects in deep neural networks. *Journal of Machine Learning Research*, 24(156):1–57, 2023.

Shiliang Sun, Jing Zhao, and Jiang Zhu. A review of nyström methods for large-scale machine learning. *Information Fusion*, 26:36–48, 2015.

Luke Tierney and Joseph B Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

Lloyd N Trefethen and David Bau. *Numerical linear algebra*, volume 181. Siam, 2022.

Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr(f(a)) via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017.

Aldo V Vecchia. Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 50(2):297–312, 1988.

Andrew Wilson and Hannes Nickisch. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). In *International conference on machine learning*, pages 1775–1784. PMLR, 2015.

Yu Yi-Sheng and Gu Dun-He. A note on a lower bound for the smallest singular value. *Linear algebra and its Applications*, 253(1-3):25–38, 1997.

Bohai Zhang, Huiyan Sang, and Jianhua Z Huang. Smoothed full-scale approximation of Gaussian process models for computation of large spatial data sets. *Statistica Sinica*, 29 (4):1711–1737, 2019.