SCALABLE GAUSSIAN PROCESS ANALYSIS FOR IMPLICIT PHYSICS-BASED COVARIANCE MODELS

Yian Chen[1,*] ***& Mihai Anitescu***[2]

[1]*Department of Statistics, University of Chicago, Chicago, IL 60637, USA*
[2]*Mathematics and Computer Science Division, Argonne National Laboratory, 9600 S. Cass Ave., Argonne, IL, 60439, USA*

Original Manuscript Submitted: mm/dd/yyyy; Final Draft Received: mm/dd/yyyy

The performance of Gaussian process analysis can be significantly affected by the choice of the covariance function. Physics-based covariance models provide a systematic way to construct covariance models that are consistent with the underlying physical laws. But the resulting model is still limited by the computational difficulties for large-scale problems. In this study, we propose a new framework combining low-rank approximations and physics-based covariance models to perform both accurate and efficient Gaussian process analysis for implicit models. The proposed approximations only interact with the physical model via a black-box forward solver and can achieve quasilinear complexity for Gaussian process regression, maximum likelihood parameter estimations, as well as approximation of the expected Fisher information matrix when performing uncertainty quantification. We also propose a way to include higher-order terms in the covariance model to account for the nonlinearities. To accomplish the goal, we choose a specific global low-rank approximation of the covariance matrix and use stochastic trace estimators. Our numerical results demonstrate the effectiveness and scalability of the approach, validate the accuracy of maximum likelihood approximations and confidence intervals, and compare the performance with other covariance models.

KEY WORDS: *Gaussian process, Low-rank approximation, Fast algorithms, Uncertainty quantification*

1. INTRODUCTION

Gaussian processes (GPs) have been widely applied in different communities for several tasks, benefiting from their non-parametric nature and accessible likelihood expression. GPs are considered standard methods of Bayesian inference [1]. For overviews of their usage, one can refer to several extensive monographs [1–3]. Gaussian process regression or kriging is used for interpolating or predicting a range of natural phenomena in geophysics and biology [3–5], taking advantage of its ability of providing fully probabilistic estimates of the uncertainty of the predictions. Extending the technique to multiple target output variables is known as co-kriging [4,6] or multi-kriging [7]. In this paper, we focus on multivariate co-kriging where the variables can be extracted from spatial processes or spatio-temporal processes.

The main computational expense of GP regression generally consists of solving lineary systems with the kernel matrix. However for an $n \times n$ dense kernel matrix, the general Cholesky decomposition approach scales as $O(n^3)$ which can easily become problematic with increasing size. Over the past decades, devising approximations to reduce the computational load has become an active research field. The approaches include active set methods [8] and partitioning methods [9,10] which separate the training data to induce sparse representation of the full problem. More significant efforts lie on efficiently obtaining low-rank approximations of the dense covariance matrix. These include global low-rank methods [11,12] which finds a compressed low-rank matrix by selecting or constructing a set of landmark points and local low-rank approximation methods by applying the framework of hierarchical matrices strategy [13,14] to obtain better time and storage complexity for certain common operations.

Another important feature of GP-based approaches is that the related probability densities can be fully characterized by their mean and covariance functions. Correct and accurate covariance models play an essential role in setting up meaningful GP regression problems. Many efforts aim to incorporate prior knowledge about the physics of the processes into the covariance model structure, which may hopefully improve the efficiency of uncertainty quantification and provide physically consistent results. A typical strategy of including prior knowledge of a real system governed by known physical laws is hierarchical Bayesian modeling [15,16]. Examples include atmospheric modeling [17–20] and environmental sciences [18,21–24]. In [25], one of the authors of this work proposed a systematic framework of assembling consistent auto- and cross-covariance models based on known physical processes. The endeavor showed that significant improvements in the forecast efficiency can be achieved by using physics-based covariance models. But some of the computational issues mentioned above for GP analysis that may appear for large datasets have nonetheless not been fully addressed.

In this work, we combine the ideas of low-rank approximation and physics-based covariance models to propose a computationally efficient, physically consistent method for identifying GP covariance parameters and carrying out the regression. We assume that the physical model is implicit, in the sense that all that is available for interacting with it is a mapping between the unknown fields (initial conditions, boundary conditions, and stochastic-noise type forcing) that can be characterized by parameterized Gaussian processes and the output. We propose an efficient approach for estimating the covariance model parameters by maximum likelihood estimation based on a low rank approximation of this mapping. Our approach achieves quasilinear complexity in the number of data points for both evaluating the log-likelihood and for carrying out the GP regression. The approach can be directly applied to linear processes. For nonlinear processes, we propose additional operations to approximate the nonlinearity with little additional cost. In this work, we aim to increase the flexibility of carrying out GP analysis by minimizing the required prior knowledge of the physical processes, providing more choices of solvers, and obtaining more control of the tradeoff between computational speed and accuracy.

The paper is organized as follows. The implicit physics-based GP regression problem is formulated and several useful techniques are reviewed in Section 2. Then scalable GP regression algorithms are developed in Section 3. Scalable maximum likelihood parameter estimations and uncertainty quantification are proposed in Section 4. Extensive numerical tests for both model simulated data and real climate data are presented in Section 5 to evaluate the effectiveness of the proposed algorithms. Finally we conclude with remarks and discussions in Section 6.

2. IMPLICIT PHYSICS-BASED COVARIANCE MODELS AND LOW-RANK APPROXIMATION

To facilitate the theoretical development of our approach, we first assume a general physical model structure to work on. Consider a general form of a deterministic model for physical processes:

$$y = F(z),\ z = (z_{x_1}, z_{x_2}, \cdots, z_{x_n})^T, \tag{1}$$

where y is an m-dimensional random field, z is an n-dimensional random field and $F \in \mathbb{R}^n \to \mathbb{R}^m$ is a sufficiently regular mapping. To distinguish the two quantities, we call y the output process (i.e. the process we can partially observe and want to predict) and z the latent process (i.e. the process that is usually not directly observed but is related to the output). We assume that the observations of y denoted by y_o and observations of z denoted by z_o are affected by observation noise

$$y_o = y + \epsilon_y,\ z_o = z + \epsilon_z \tag{2}$$

where ϵ_y, ϵ_z are independent Gaussian processes with covariance $K_{\epsilon_y} = \mathrm{Cov}(\epsilon_y, \epsilon_y)$ and $K_{\epsilon_z} = \mathrm{Cov}(\epsilon_z, \epsilon_z)$ respectively. Note that our framework allows multiple output components and multiple components in the latent process. Each component may have different levels of observation noise due to their possibly different physical nature, measurements, units, and conditions. We assume the observation noise covariance K_{ϵ_y} and K_{ϵ_z} can be efficiently inverted with at most quasilinear time cost. For example in a lot of cases, such as when each component is measured by a different instrument, the noise processes ϵ_y and ϵ_z are component-wise independent and can many times be assumed to be Gaussian [26]. Then their covariance matrix is only a diagonal matrix with entries representing different noise magnitudes and it is easily invertible with linear time cost. We thus find that our assumptions include a broad set of models.

We also note that many models can be adapted or converted to this form. For example, many physical processes can be modeled using differential equations or stochastic differential equations. Additional constraints such as initial conditions and boundary conditions are needed in order to solve these equations. In this case, the model can be expressed by

$$y = G(B, I, \omega), \tag{3}$$

where B denotes the boundary condition, I denotes the initial condition and ω is the random process component of the stochastic differential equation. To convert the model into general form, we can concatenate the latent variables into a column vector of latent process $z^T = (B^T, I^T, \omega^T)$. Note that we will use the general form (1) in the theoretical derivation of this study, however form (3) may be simpler to use in applicable cases since separating independent variables may provide a more compact solution and reduce the size of the problem.

2.1 Physics-based Covariance Models

Gaussian process regression requires good covariance models which can accurately capture the joint variability of the processes. By incorporating information from the physical model (1), we use physics-based covariance models proposed in [25]. Note that although [25] provides covariance models only for spatial processes, they can be extended to spatio-temporal context with little extra effort.

2.1.1 Second-order Covariance Model

Denote the expectation $\mathrm{E}(z) = \bar{z}$ and the small perturbation around the mean value by $\delta z = z - \bar{z}$. If two processes satisfy the physical constraint given by (1),(2) for $F \in \mathcal{C}^2$, the covariance matrix formed by y_o and z satisfies

$$\mathrm{Cov}\left(\begin{bmatrix} y_o \\ z \end{bmatrix}\right) = \begin{bmatrix} L\mathrm{Cov}(z,z)L^T + K_{\epsilon_y} & L\mathrm{Cov}(z,z) \\ \mathrm{Cov}(z,z)L^T & \mathrm{Cov}(z,z) \end{bmatrix} + O(||\delta z||^3), \tag{4}$$

where L is the Jacobian matrix of F evaluated at $\bar{z}$. If we use the matrix above as the estimate of the joint covariance, we would incur a third-order error in δz. If the function F is linear, the error term vanishes and the covariance model is exact. This covariance model can be applied when the corresponding physical process can be exactly or approximately well represented by its linearization. If the function is highly nonlinear, extra higher-order terms can be added to the covariance model to account for the nonlinearity and lead to more accurate results.

2.1.2 Higher-order Covariance Model

To use higher-order expansions, we will need to compute tensor operations. For further detail on the tensor algebra we use, see APPENDIX B. In particular, note that for H, the Hessian tensor of $F(z)$ defined in (1), a vector $v \in \mathbb{R}^n$, we have that $v^T H v$ is a column vector, the transpose of which is the row vector $v^T H^T v$.

With these conventions, the covariance model under higher-order closure is given by [25]

$$\mathrm{Cov}\left(\begin{bmatrix} y_o \\ z \end{bmatrix}\right) = \begin{bmatrix} K_{11} + K_{\epsilon_y} & L\mathrm{Cov}(z,z) \\ \mathrm{Cov}(z,z)L^T & \mathrm{Cov}(z,z) \end{bmatrix} + O(||\delta z||^4), \tag{5}$$

where L is the Jacobian matrix, H is the Hessian tensor of $F(z)$ evaluated at $\bar{z}$, and [25]

$$K_{11} = L\mathrm{Cov}(z,z)L^T + \frac{1}{4}\overline{\delta z^T H \delta z \delta z^T H^T \delta z} - \frac{1}{4}\overline{\delta z^T H \delta z}\;\overline{\delta z^T H^T \delta z}. \tag{6}$$

This covariance model includes terms up to order four. If the function F is quadratic, the error term vanishes and the covariance model is exact. For more complicated processes, the truncated covariance model (5) can be used as a good approximation of the exact covariance matrix. One fact worth noting is that the higher-order terms only appear in the auto-covariance. The cross-covariance remains the same under both second-order and higher-order closure assumption.

2.2 Low-rank Approximation of Kernel Using Chebyshev Interpolation

Carrying out GP regression in the general covariance case requires computing the inverse of, or at least solving linear systems with, large, dense covariance matrices. Conventional methods based on Cholesky decomposition are expensive since the computational cost scales as $O(n^3)$ for an $n \times n$ dense matrix. In order to obtain a computationally efficient method for solving the GP regression problem, we use low-rank approximation methods that facilitate linear algebra with better scaling. In this study, we approximate the covariance matrix of the latent process $\mathrm{Cov}(z, z)$ by Chebyshev interpolation. Note that low-rank kernel approximation using other choices of interpolation methods and nodes is also possible; we refer readers to [27] for a discussion. If the Gaussian process describing z is smooth enough, the Chebyshev interpolation is sufficient for our purpose. In some circumstances, particularly at high sample density, using smooth covariance models may not be a suitable representation of the uncertainty [4], and in that case our approach would not produce a good approximation. Our approach can be relatively easily extended to a block diagonal plus low rank structure of $\mathrm{Cov}(z, z)$ if we make some assumptions about the resulting covariance, or by using the fact that for many applications L itself is low rank [28] to allow for cases where the covariance is rougher. For this paper we restrict ourselves to the smooth $\mathrm{Cov}(z, z)$ which we will show can have a positive impact for some observed data problems in §5.

Chebyshev nodes are roots of Chebyshev polynomial of the first kind. Using Chebyshev nodes as interpolation points in polynomial interpolation (Chebyshev interpolation) can help minimize the effect of Runge's phenomenon. Given the number of interpolation points N, in the interval $(-1, 1)$, the Chebyshev nodes are defined by

$$\tilde{x}_k = \cos\left(\frac{2k-1}{2N}\pi\right),\ k = 1, 2, \cdots, N. \tag{7}$$

For an 1D smooth function $f : (-1, 1) \to \mathbb{R}$, given function evaluations at the set of interpolation points $\{(\tilde{x}_1, f(\tilde{x}_1)), (\tilde{x}_2, f(\tilde{x}_2)), \cdots, (\tilde{x}_N, f(\tilde{x}_N))\}$, then for any $x \in (-1, 1)$, $f(x)$ can be approximated using Lagrange polynomials,

$$f(x) = \sum_{i=1}^{N}\left(\prod_{j\neq i}\frac{x-\tilde{x}_j}{\tilde{x}_i-\tilde{x}_j}\right) f(\tilde{x}_i). \tag{8}$$

The approach can be generalized to functions over arbitrary interval domains by affine transformation. Now our goal is to construct a low-rank compressed kernel matrix K, from which the kernel matrix $\mathrm{Cov}(z, z)$ can be approximately reconstructed by Chebyshev interpolation. Assume $z(x) = (z(x_1), z(x_2), \cdots, z(x_n))^T$ is a random field over an n-dimensional location set $(x_1, x_2, \cdots, x_n)$, we can find N Chebyshev nodes $(\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_N)$ over the location set. Define the compressed covariance matrix $K \in \mathbb{R}^{N\times N}$ by

$$K = (K_{ij}),\ K_{ij} = \mathrm{Cov}(z(\tilde{x}_i), z(\tilde{x}_j)),\ i, j = 1, 2, \cdots, N. \tag{9}$$

In addition, define coefficient vector associated with x by

$$c(x) = \left(\prod_{j\neq 1}\frac{x-\tilde{x}_j}{\tilde{x}_1-\tilde{x}_j}, \prod_{j\neq 2}\frac{x-\tilde{x}_j}{\tilde{x}_2-\tilde{x}_j}, \cdots, \prod_{j\neq N}\frac{x-\tilde{x}_j}{\tilde{x}_N-\tilde{x}_j}\right)^T \in \mathbb{R}^N. \tag{10}$$

Construct $C_z = (c(x_1), c(x_2), \cdots, c(x_n)) \in \mathbb{R}^{N\times n}$. Then we can approximate $\mathrm{Cov}(z, z)$ by $C_z^T K C_z$, which is of rank N. If we take $N < n$, it becomes a valid low-rank approximation of $\mathrm{Cov}(z, z)$. Note that the same Chebyshev interpolation can be easily extended to multiple dimensions using tensor products. It can be useful when the latent process has more than one dimension.

2.3 Approximation of Jacobian and Hessian

As we have noticed, the physics-based covariance model requires computing the Jacobian and Hessian matrices of the function F at given points. Since the physical model is implicit, we have no algebraic expression of F and the information of the model is only available via some black-box forward solvers. To avoid direct estimation and storage of the Jacobian and Hessian, we use centered difference scheme to efficiently approximate the Jacobian-vector product and the vector-Hessian-vector product for any vectors. While automatic differentiation is always a possibility, when one works with complex models, as, for example, climate codes, getting automatic differentiation to work is a complex and time-consuming endeavor [29]. Our simple approach has downsides, including the one of introducing additional approximation errors, but it is highly parallelizable. Here are the approximations we use.

- Jacobian-vector product Lu for vector $u \in \mathbb{R}^n$

Let s be a sufficiently small positive real number, then by Taylor expansion, we can approximate Lu by

$$Lu = \frac{1}{2s}(F(\bar{z}+su) - F(\bar{z}-su)) + O(s^2||u||^3). \tag{11}$$

- Vector-Hessian-vector product $u^T Hv$ for vector $u, v \in \mathbb{R}^n$

Let $L(z)$ be the Jacobian matrix of F evaluated at z. Let r be a sufficiently small positive real number. Consider the Taylor expansion of $L(\bar{z}+rv)$ and $L(\bar{z}-rv)$,

$$L(\bar{z}+rv)u = L(\bar{z})u + rv^T Hu + O(r^3||v||^3), \tag{12}$$
$$L(\bar{z}-rv)u = L(\bar{z})u - rv^T Hu + O(r^3||v||^3). \tag{13}$$

Subtracting the two equations, we get the approximation

$$v^T Hu = \frac{1}{2r}(L(\bar{z}+rv)u - L(\bar{z}-rv)u) + O(r^2||v||^3). \tag{14}$$

Combine with the approximation of Jacobian-vector product (11), we get

$$\begin{aligned} v^T Hu =& \frac{1}{4rs}[F(\bar{z}+rv+su) - F(\bar{z}+rv-su) - F(\bar{z}-rv+su) + F(\bar{z}-rv-su)] \\ &+ O(r^2||v||^3 + \frac{s^2||u||^3}{r}). \end{aligned} \tag{15}$$

For both Jacobian and Hessian approximations, when $s, r \to 0$, and $\frac{s}{r}$ is bounded above, the error tends to zero. In practice, we choose s, r small and of the same order to make su, rv be small perturbations around $\bar{z}$.

3. A SCALABLE APPROACH FOR GAUSSIAN PROCESS REGRESSION USING PHYSICS-BASED COVARIANCE MODELS

The GP regression problem aims to interpolate the whole field y by partial observations. Specifically, we can partition the whole field y into an observed part and an unobserved part. Assume the partial observations (with observation noise) of y are denoted by $y_o \in \mathbb{R}^d$ and the remaining unobserved field which we want to predict is denoted by $y_p \in \mathbb{R}^{m-d}$. Assume y satisfies the physical constraint (1), then we can correspondingly partition the vector-valued function F by

$$y_o = F_o(z) + \epsilon_y, \tag{16}$$
$$y_p = F_p(z). \tag{17}$$

We consider the following two scenarios: (a) we only have observations of y in §3.1 and (b) we have additional observations of the latent process z in §3.2. The name of the resulting tasks, latent process kriging (a) and joint process kriging (b) originate in [25] when applied to the explicit covariance model. We now describe them and we evaluate their computational effort as a combination of number of forward model evaluations required by the derivations in §2.3, number of system solves with the covariance matrices defining our model (2), and the (dominant) computational effort of the remaining model forming tasks.

3.1 Latent Process Kriging

By the physics-based covariance model (4), the joint distribution of y_o and y_p is approximated by

$$\mathcal{M}_1 : \begin{pmatrix} y_o \\ y_p \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(y_o) \\ m(y_p) \end{pmatrix}, \begin{pmatrix} L_o \mathrm{Cov}(z,z) L_o^T + K_{\epsilon_y} & L_o \mathrm{Cov}(z,z) L_p^T \\ L_p \mathrm{Cov}(z,z) L_o^T & L_p \mathrm{Cov}(z,z) L_p^T \end{pmatrix} \right), \tag{18}$$

where L_o is the Jacobian matrix of F_o evaluated at $\bar{z}$ and L_p is the Jacobian matrix of F_p evaluated at $\bar{z}$. The corresponding GP regression problem solves the posterior distribution under observations. The predicted posterior mean is given by

$$m(y_p) + (L_p \mathrm{Cov}(z,z) L_o^T)(K_{\epsilon_y} + L_o \mathrm{Cov}(z,z) L_o^T)^{-1}(y_o - m(y_o)). \tag{19}$$

The difficulty of computing the expression is that $(K_{\epsilon_y} + L_o \mathrm{Cov}(z,z) L_o^T)$ is a $d \times d$ matrix. Solving the inverse by conventional Cholesky decomposition method requires a computational cost of order $O(d^3)$. Since we assume K_{ϵ_y} possesses sparsity and can be efficiently inverted, the Sherman-Morrison-Woodbury (SMW) formula can then be applied and the computational cost of solving the inverse is then determined by the rank of $L_o \mathrm{Cov}(z,z) L_o^T$ term, i.e. $\max(d,n)$. For simplicity of our discussion, we assume $d = O(m)$, which models many circumstances where the dimension of the vector to be forecasted is no larger in order than the number of observation points required. Then the computational cost can be reduced by the proposed low-rank approximation, which replaces $L_o \mathrm{Cov}(z,z) L_o^T$ by a rank N approximation. We illustrate the scalable approach for solving the GP regression posterior mean with computational cost analysis of each step in Algorithm 1.

In summary, based on our assumption of observation noise covariance K_{ϵ_y}, the entire approach takes $O(N)$ forward solves, $O(N)$ K_{ϵ_y} linear system solves and the dominant computational cost takes $O(mN^2 + nN)$ time, though the workflow is highly parallelizable. The cost is quasilinear with n if $N = log(n)$.

3.2 Joint Process Kriging

In this subsection, we consider the case where we can partially observe the latent process. The additional information about the latent process helps us get more accurate predictions of the unobserved output. Denote the observed part of the latent process by $z_o \in \mathbb{R}^D$ and the unobserved part of latent process by $z_p \in \mathbb{R}^{n-D}$. The observation noise covariance is given by K_{ϵ_z}. As discussed after (2), we assume K_{ϵ_z} is sparse and can be efficiently inverted with at most quasilinear time cost. For computational effort estimation purposes we assume $D = O(n)$. From our physics-based covariance model (4), the joint distribution of y_o, z_o, y_p and z_p is given by

$$\mathcal{M}_2 : \begin{pmatrix} y_o \\ z_o \\ y_p \\ z_p \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(y_o) \\ m(z_o) \\ m(y_p) \\ m(z_p) \end{pmatrix}, \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix} \right), \tag{23}$$

where

$$K_{11} = \begin{pmatrix} K_{\epsilon_y} + L_o \mathrm{Cov}(z,z) L_o^T & L_o \mathrm{Cov}(z, z_o) \\ \mathrm{Cov}(z_o, z) L_o^T & K_{\epsilon_z} + \mathrm{Cov}(z_o, z_o) \end{pmatrix}, \tag{24}$$

```
Algorithm 1: Latent Process Kriging $\mathcal{M}_1$
Use the low-rank approximation discussed in §2.2 with $N = O(\log n)$ and then solve the approximate GP
  regression,
    $m(y_p) + (L_p C_z^T K C_z L_o^T)(K_{\epsilon_y} + L_o C_z^T K C_z L_o^T)^{-1}(y_o - m(y_o)).$    (20)
Step 1. Compute $A_1 = L_o C_z^T \in \mathbb{R}^{d\times N}$, $A_2 = L_p C_z^T \in \mathbb{R}^{(m-d)\times N}$ by $O(N)$ forward solves.
Step 2. Solve the approximated inverse problem by the SMW formula.
    $(K_{\epsilon_y} + A_1 K A_1^T)^{-1}(y_o - m(y_o))$
    $= K_{\epsilon_y}^{-1}(y_o - m(y_o)) - K_{\epsilon_y}^{-1} A_1 (K^{-1} + A_1^T K_{\epsilon_y}^{-1} A_1)^{-1} A_1^T K_{\epsilon_y}^{-1}(y_o - m(y_o)).$    (21)
    a. Compute $\alpha = A_1^T K_{\epsilon_y}^{-1}(y_o - m(y_o))$. It takes $O(mN)$ time and one $K_{\epsilon_y}$ linear system solve.
    b. Solve $(K^{-1} + A_1^T K_{\epsilon_y}^{-1} A_1)^{-1}\alpha$ by
        $\begin{pmatrix} I_N & A_1^T K_{\epsilon_y}^{-1} A_1 \\ K & -I_N \end{pmatrix}\begin{pmatrix}\beta \\ \gamma\end{pmatrix} = \begin{pmatrix}\alpha \\ 0\end{pmatrix},$    (22)
       where forming $A_1^T K_{\epsilon_y}^{-1} A_1$ takes $O(N)$ $K_{\epsilon_y}$ linear system solves and $O(mN^2)$ time. Solving the
       system takes $O(N^3)$ time.
    c. The solution is given by $y = K_{\epsilon_y}^{-1}(y_o - m(y_o)) - K_{\epsilon_y}^{-1} A_1 \gamma$. This takes $O(mN)$ time and two $K_{\epsilon_y}$
       linear system solves.
Step 3. Return $m(y_p) + (A_2 K A_1^T) y$ by matrix-vector multiplication. This takes $O(2mN + N^2)$ time.
```

$$K_{21} = \begin{pmatrix} L_p \mathrm{Cov}(z,z) L_o^T & L_p \mathrm{Cov}(z, z_o) \\ \mathrm{Cov}(z_p, z) L_o^T & \mathrm{Cov}(z_p, z_o) \end{pmatrix}. \tag{25}$$

The corresponding GP regression problem solves the predicted posterior mean,

$$\begin{pmatrix} m(y_p) \\ m(z_p) \end{pmatrix} + (K_{21})(K_{11})^{-1}\begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o) \end{pmatrix}. \tag{26}$$

The partitioned covariance matrices $\mathrm{Cov}(z_o, z)$, $\mathrm{Cov}(z_p, z)$, $\mathrm{Cov}(z_p, z_o)$ are all submatrices of $\mathrm{Cov}(z, z)$ formed by selecting the corresponding rows or columns. Therefore, we can construct $C_{z_o} \in \mathbb{R}^{N\times D}$ and $C_{z_p} \in \mathbb{R}^{N\times(n-D)}$ by selecting corresponding columns of the interpolation matrix C_z and creating suitable low rank approximations of these covariance matrices. A scalable approach of solving the GP regression posterior mean problem is summarized in Algorithm 2.

In summary, based on our assumption of observation noise covariance matrices, the entire approach takes $O(N)$ forward solves, $O(N)$ observation noise covariance matrix linear system solves and the dominant computational cost takes $O((m+n)N^2)$ time. The cost is quasilinear with n if $N = log(n)$.

3.3 Correction for Nonlinearity

In the previous two subsections, we described scalable approaches for solving the GP regression problem with physics-based second-order covariance model (4). For highly nonlinear functions, we expect that using higher-order covariance models (5) will provide more accurate covariance matrix structure. In this subsection, we will discuss how to include the higher-order terms in GP regression problem while maintaining quasilinear time scaling via approximations.

To this end, we apply the formalism (6) to the partitioned case where $y_o \to ((y_o)^T, yP^T)^T$. We denote the Hessian tensor of F_o, F_p evaluated at $\bar{z}$ by H_o, H_p respectively. By expression (6), one important observation is that

Algorithm 2: Joint Process Kriging $\mathcal{M}_2$

Use low-rank approximation index $N = O(\log n)$. Approximate the latent process covariances by interpolation as $\mathrm{Cov}(z_o, z_o) \approx C_{z_o}^T K C_{z_o}$, $\mathrm{Cov}(z_o, z) \approx C_{z_o}^T K C_z$, $\mathrm{Cov}(z_p, z) \approx C_{z_p}^T K C_z$, $\mathrm{Cov}(z_p, z_o) \approx C_{z_p}^T K C_{z_o}$.

Step 1. Compute $A_1 = L_o C_z^T \in \mathbb{R}^{d\times N}$, $A_2 = L_p C_z^T \in \mathbb{R}^{(m-d)\times N}$ by $O(N)$ forward solves.

Step 2. Solve the approximated inverse problem by the SMW formula,

$$\begin{aligned}\alpha =&(K_{11})^{-1} \begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o) \end{pmatrix} \\ =& \left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z} \end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T \end{pmatrix} \begin{pmatrix} K & K \\ K & K \end{pmatrix} \begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o} \end{pmatrix} \right)^{-1} \begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o) \end{pmatrix}. \end{aligned} \tag{27}$$

This can be efficiently carried out since $\begin{pmatrix} K & K \\ K & K \end{pmatrix} \in \mathbb{R}^{2N\times 2N}$ has dimension much smaller than n, and $\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z} \end{pmatrix}$ can be efficiently inverted similarly to Algorithm 1. The time cost is dominated by forming matrix products $(A_1^T K_{\epsilon_y}^{-1} A_1)$ and $(C_{z_o} K_{\epsilon_z}^{-1} C_{z_o}^T)$ which takes $O((n+m)N^2)$ time and $O(N)$ linear system solves with K_{ϵ_y} and K_{ϵ_z}.

Step 3. **Return** the final solution,

$$\begin{pmatrix} m(y_p) \\ m(z_p) \end{pmatrix} + \left(\begin{pmatrix} A_2 & 0 \\ 0 & C_{z_p}^T \end{pmatrix} \begin{pmatrix} K & K \\ K & K \end{pmatrix} \begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o} \end{pmatrix} \right) \alpha. \tag{28}$$

The step takes $O(2(n+m)N + 4N^2)$ time.

the higher-order terms only occur in the auto-covariance of y. Our modified latent process kriging with higher-order terms then has the following expression,

$$\begin{aligned} m(y_p) + & \left(L_p \mathrm{Cov}(z,z) L_o^T + \frac{1}{4} \overline{\delta z^T H_p \delta z \delta z^T H_o^T \delta z} - \frac{1}{4} \overline{\delta z^T H_p \delta z} \right) \overline{\delta z^T H_o^T \delta z}) \\ & \left(K_{\epsilon_y} + L_o \mathrm{Cov}(z,z) L_o^T + \frac{1}{4} \overline{\delta z^T H_o \delta z \delta z^T H_o^T \delta z} - \frac{1}{4} \overline{\delta z^T H_o \delta z}\ \overline{\delta z^T H_o^T \delta z} \right)^{-1} (y_o - m(y_o)). \end{aligned} \tag{29}$$

Likewise, for joint process kriging with higher-order terms, the estimator becomes

$$\begin{pmatrix} m(y_p) \\ m(z_p) \end{pmatrix} + (K_{21}^+)(K_{11}^+)^{-1} \begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o) \end{pmatrix}. \tag{30}$$

where

$$K_{11}^+ = \begin{pmatrix} K_{\epsilon_y} + L_o \mathrm{Cov}(z,z) L_o^T + \frac{1}{4} \overline{\delta z^T H_o \delta z \delta z^T H_o^T \delta z} - \frac{1}{4} \overline{\delta z^T H_o \delta z}\ \overline{\delta z^T H_o^T \delta z} & L_o \mathrm{Cov}(z, z_o) \\ \mathrm{Cov}(z_o, z) L_o^T & K_{\epsilon_z} + \mathrm{Cov}(z_o, z_o) \end{pmatrix}, \tag{31}$$

$$K_{21}^+ = \begin{pmatrix} L_p \mathrm{Cov}(z,z) L_o^T + \frac{1}{4} \overline{\delta z^T H_p \delta z \delta z^T H_o^T \delta z} - \frac{1}{4} \overline{\delta z^T H_p \delta z}\ \overline{\delta z^T H_o^T \delta z} & L_p \mathrm{Cov}(z, z_o) \\ \mathrm{Cov}(z_p, z) L_o^T & \mathrm{Cov}(z_p, z_o) \end{pmatrix}. \tag{32}$$

We now focus on computing the two higher-order terms appearing in auto-covariance. Denote

$$U(H,\hat{H}) = \frac{1}{4}\overline{\delta z^T H \delta z \delta z^T \hat{H}^T \delta z}, \tag{33}$$

$$V(H,\hat{H}) = -\frac{1}{4}\overline{\delta z^T H \delta z}\;\overline{\delta z^T \hat{H}^T \delta z}. \tag{34}$$

where $H, \hat{H}$ both take value of either H_o or H_p, depending on the covariance components. For multivariate Gaussian random variable δz, we have that

$$\begin{aligned}
U(H,\hat{H}) + V(H,\hat{H}) &\overset{\text{(C.9),(C.11)}}{=} \frac{1}{4}\overline{\delta z^T H \delta z \delta z^T \hat{H}^T \delta z} - \frac{1}{4}\overline{\delta z^T H \delta z}\;\overline{\delta z^T \hat{H}^T \delta z} \\
&= \frac{1}{2}\mathrm{tr}(H\mathrm{Cov}(z,z)\hat{H}^T\mathrm{Cov}(z,z)) + \frac{1}{4}\mathrm{tr}(H\mathrm{Cov}(z,z))\mathrm{tr}(\hat{H}\mathrm{Cov}(z,z))^T \\
&\quad - \frac{1}{4}\mathrm{tr}(H\mathrm{Cov}(z,z))\mathrm{tr}(\hat{H}\mathrm{Cov}(z,z))^T \\
&= \frac{1}{2}\mathrm{tr}(H\mathrm{Cov}(z,z)\hat{H}^T\mathrm{Cov}(z,z)).
\end{aligned} \tag{35}$$

Details of the algebra can be found in APPENDIX C.

We now turn to the issue of approximating the nonlinear corrections efficiently. We will again use the low-rank approximation $\mathrm{Cov}(z,z) \approx C_z^T K C_z$ with K symmetric and positive semi-definite. Then there exists a Cholesky factorization of K such that $K = P^T P$, with P is an upper triangular matrix. Since K is an $N \times N$ matrix, with $N \ll n$, the cost of the factorization can be ignored comparing to quasilinear scaling in n. Our approximation becomes

$$\begin{aligned}
\mathrm{tr}(H\mathrm{Cov(z,z)}\hat{H}^T\mathrm{Cov(z,z)}) &\approx \mathrm{tr}(HC_z^T P^T P C_z \hat{H}^T C_z^T P^T P C_z) \\
&= \mathrm{tr}(PC_z H C_z^T P^T P C_z \hat{H}^T C_z^T P^T) \\
&= \mathrm{tr}(M\hat{M}^T),
\end{aligned} \tag{36}$$

where $M = PC_z H C_z^T P^T$, $\hat{M} = PC_z \hat{H} C_z^T P^T$ are rank-three tensors, whose size depends on the choice of Hessian tensors $H, \hat{H}$. We further denote $M_o = PC_z H_o C_z^T P^T$, $M_p = PC_z H_p C_z^T P^T$ corresponding to the two choices of Hessians. Let $\{u_i\}, \{v_i\}$ be N-dimensional independent Rademacher vectors (random vectors with independent identical Bernoulli distributed components that take the values $+1$ and -1 with probability 0.5), then

$$\begin{aligned}
\mathrm{tr}(M\hat{M}^T) &= \mathrm{tr}(MI\hat{M}^T I) \\
&= \mathrm{tr}(M\mathrm{E}(uu^T)\hat{M}^T\mathrm{E}(vv^T)) \\
&= \mathrm{E}(\mathrm{tr}(Muu^T\hat{M}^T vv^T)) \\
&= \mathrm{E}(v^T M u u^T \hat{M}^T v) \\
&\approx \frac{1}{N_l}\sum_{i=1}^{N_l}(v_i^T M u_i)(v_i^T \hat{M} u_i)^T.
\end{aligned} \tag{37}$$

Note that $v_i^T M u_i \in \mathbf{R}^{m_2}$ where m_2 is the second dimension of M, and $v_i^T M u_i \in \mathbf{R}^{\hat{m}_2}$ where $\hat{m}_2$ is the second dimension of $\hat{M}$. Therefore $(v_i^T M u_i)(v_i^T \hat{M} u_i)^T$ is a rank-1 $m_2 \times \hat{m}_2$ matrix. The entire stochastic trace estimator (37) is a matrix of rank-N_l. When $N_l = O(\log n)$, adding the approximated trace term to the auto-covariance matrices becomes an additional low-rank perturbation which can also be handled efficiently by the SMW formula. While there is a concern that N_l may be too small to obtain a high quality approximation, we note that for well conditioned blocks $M(:,i,:)$, for $i = 1, 2, \ldots, m_2$ (in the Matlab notation), the approximation has provably excellent quality [30]. Since

Algorithm 3: Latent Process Kriging with Higher-order terms $\mathcal{M}_1^+$

Use the low-rank approximation with $N = O(\log n)$ and then solve the approximated problem,

$$m(y_p) + \left(L_p C_z^T K C_z L_o^T + \frac{1}{N_l} \sum_{i=1}^{N_l} (v_i^T M_p u_i)(v_i^T M_o u_i)^T \right) \left(K_{\epsilon_y} + L_o C_z^T K C_z L_o^T + \frac{1}{N_l} \sum_{i=1}^{N_l} (v_i^T M_o u_i)(v_i^T M_o u_i)^T \right)^{-1} (y_o - m(y_o)). \tag{38}$$

Step 1. Compute $A_1 = L_o C_z^T \in \mathbb{R}^{d\times N}$, $A_2 = L_p C_z^T \in \mathbb{R}^{(m-d)\times N}$ by $O(N)$ forward solves.

Step 2. Compute the Cholesky factorization $K = P^T P$. This takes $O(N^3)$ time.

Step 3. Draw $2N_l$ independent Rademacher vectors $\{u_i\}, \{v_i\}$. Compute

$$\phi_i = C_z^T P^T u_i, \psi_i = C_z^T P^T v_i. \tag{39}$$

This takes $O(2N_l(N^2 + nN)))$ time.

Step 4. Compute the vector-Hessian-vector product $w_i = \psi_i^T H_o \phi_i$, $w_i' = \psi_i^T H_p \phi_i$, $i = 1, 2, \dots, N_l$, by the approximation (17). This approximation takes $O(N_l)$ forward solves. Define the matrices $A_3 \leftarrow \frac{1}{\sqrt{2N_l}}(w_1, w_2, \cdots, w_{N_l}) \in \mathbb{R}^{d\times N_l}$, $A_4 \leftarrow \frac{1}{\sqrt{2N_l}}(w_1', w_2', \cdots, w_{N_l}') \in \mathbb{R}^{(m-d)\times N_l}$. From (37) we will approximate the higher-order terms by

$$\frac{1}{2}\mathrm{tr}(H_o \mathrm{Cov}(z, z) H_o^T \mathrm{Cov}(z, z)) \approx A_3 A_3^T, \tag{40}$$

$$\frac{1}{2}\mathrm{tr}(H_p \mathrm{Cov}(z, z) H_o^T \mathrm{Cov}(z, z)) \approx A_4 A_3^T. \tag{41}$$

Step 5. Solve the modified inverse problem

$$\begin{aligned} \alpha &= (K_{\epsilon_y} + A_1 K A_1^T + A_3 A_3^T)^{-1}(y_o - m(y_o)). \\ &= \left(K_{\epsilon_y} + \begin{pmatrix} A_1 P^T & A_3 \end{pmatrix} \begin{pmatrix} P A_1^T \\ A_3^T \end{pmatrix} \right)^{-1} (y_o - m(y_o)) \end{aligned} \tag{42}$$

by applying the SMW formula for the rank-$(N + N_l)$ perturbation, same as in Algorithm 1. The time cost is dominated by computing the matrix products that define the blocks of $\begin{pmatrix} A_3^T K_{\epsilon_y}^{-1} A_3 & A_3^T K_{\epsilon_y}^{-1} A_1 P^T \\ P A_1^T K_{\epsilon_y}^{-1} A_3 & P A_1^T K_{\epsilon_y}^{-1} A_1 P^T \end{pmatrix}$.
This takes $O(N + N_l)$ linear system solves with K_{ϵ_y} and $O(m(N + N_l)^2)$ time.

Step 6. **Return** $m(y_p) + (A_2 K A_1^T + A_4 A_3^T)\alpha$ by matrix-vector product. It takes $O(2m(N + N_l) + N^2)$ time.

a low rank approximation of a covariance matrix has a much better condition number than the original, the approach is in the regime of validity from [30].

We illustrate the computational workflow for including nonlinear higher-order terms in the covariance computation and solving the modified GP regression problem scalably in Algorithm 3. The similar workflow for the joint process kriging is in APPENDIX A.

In summary, based on our assumption of observation noise covariance matrices, the entire approach takes $O(N + N_l)$ forward solves, $O(N + N_l)$ observation noise covariance matrix linear system solves, and the dominant remaining computational cost takes $O(m(N + N_l)^2)$ time. The cost is quasilinear with n if $N, N_l = log(n)$.

4. A SCALABLE APPROACH FOR GAUSSIAN PROCESS MAXIMUM LIKELIHOOD ESTIMATION

The proposed physics-based covariance model (3)(4) provides explicit covariance structure based on the physical operators and the covariance of the latent process $\mathrm{Cov}(z, z)$. In most applications, a reasonable covariance function of the latent process can be proposed based on past observations or prior knowledge up to some parameters, that need to be estimated from data. Since z is a random field at locations $(x_1, \cdots, x_n)$, assume the covariance matrix $\mathrm{Cov}(z, z)$ can be parameterized by a valid covariance function $k(\cdot, \cdot; \theta)$ depending on unknown parameters $\theta \in \mathbb{R}^r$. Then

$$\mathrm{Cov}(z(x_i), z(x_j); \theta) = k(x_i, x_j; \theta),\ i, j = 1, 2, \cdots, n. \tag{43}$$

We denote the resulting covariance matrix by $\mathrm{Cov}(z, z; \theta) \in \mathbb{R}^{n\times n}$. The most principled method to estimate the parameters θ is the maximum likelihood estimator (MLE). Following the same setup we distinguish two cases: (a) a latent process MLE, where we only have available observations of output process y and (b) a joint process MLE, where we can observe both processes y and z, which we describe below. Notionally, the same estimation workflow could apply to the nonlinear corrections from §3.3, but the estimates can no longer be justified by output likelihood considerations, since in the case of nonlinear transformation the output y does not have a Gaussian distribution in general. We thus discuss here only the linear dependence case. Moreover y_p does not play a role in estimating θ, so we will focus to the joint relationship of y_o and z only.

4.1 Latent Process MLE

Given partial observations $y_o \in \mathbb{R}^d$ and based on the covariance model for the latent process (18), the log-likelihood function (up to an additive constant) is given by

$$\log p(y_o|\theta) = -\frac{1}{2}\log\det\left(K_{\epsilon_y} + L_o\mathrm{Cov}(z, z; \theta)L_o^T\right) - \frac{1}{2}(y_o - \overline{y_o})^T(K_{\epsilon_y} + L_o\mathrm{Cov}(z, z; \theta)L_o^T)^{-1}(y_o - \overline{y_o}). \tag{44}$$

One can avoid the log-determinant computation – which requires a Cholesky factorization of a dense and large matrix, a very expensive computation – by considering the score equations, which are obtained by setting the gradient of log-likelihood to be zero. The gradient components are:

$$\begin{aligned} g_i(\theta) =& \frac{1}{2}(y_o - \overline{y_o})^T(K_{\epsilon_y} + L_o\mathrm{Cov}(z, z; \theta)L_o^T)^{-1}(L_o\mathrm{Cov}_i(z, z; \theta)L_o^T)(K_{\epsilon_y} + L_o\mathrm{Cov}(z, z; \theta)L_o^T)^{-1}(y_o - \overline{y_o}) \\ &- \frac{1}{2}\mathrm{tr}((K_{\epsilon_y} + L_o\mathrm{Cov}(z, z; \theta)L_o^T)^{-1}(L_o\mathrm{Cov}_i(z, z; \theta)L_o^T)), \end{aligned} \tag{45}$$

where $\mathrm{Cov}_i(z, z; \theta) = \frac{\partial}{\partial\theta_i}\mathrm{Cov}(z, z; \theta)$. The score equations become $g_i(\theta) = 0$, $i = 1, 2, \ldots, r$. To solve this problem, one needs to be able to efficiently evaluate the score equation for any θ. When creating low-rank approximations, we exploit the fact that the Chebyshev nodes and coefficients do not depend on the parameters θ. Using the low-rank approximation discussed in Section 2.2, we can define the following approximation

$$\mathrm{Cov}(z, z; \theta) \approx C_z^T K(\theta) C_z, \tag{46}$$

$$\frac{\partial}{\partial\theta_i}\mathrm{Cov}(z, z; \theta) \approx C_z^T K_i(\theta) C_z,\quad i = 1, 2, \ldots, r \tag{47}$$

where $K(\theta)$ is the compressed covariance matrix, explicitly depending on θ and $K_i(\theta)$ its derivative with θ_i. Due to the explicit nature of $K(\theta)$ in an interpolation approach, $K_i(\theta)$ is readily computable. Therefore, the two terms in the gradient component expression (45) can be approximated separately. For the first term, we apply the SMW formula to sequentially solve the matrix-vector product. For the second term, we use stochastic trace estimator to convert the expensive trace operations to vector-matrix-vector product [30]. A scalable approach for solving latent process MLE problem is summarized in Algorithm 4.

```
Algorithm 4: Latent Process Score equations
  Step 1. Compute $A_1 = L_o C_z^T \in \mathbb{R}^{d\times N}$ using $O(N)$ forward solves.
  Step 2. Compute the approximation of the first term in the score equations,
      $\alpha_1^i \approx \frac{1}{2}(y_o - \overline{y_o})^T (K_{\epsilon_y} + A_1 K(\theta) A_1^T)^{-1} (A_1 K_i(\theta) A_1^T)(K_{\epsilon_y} + A_1 K(\theta) A_1^T)^{-1}(y_o - \overline{y_o})$   (48)
      $= \frac{1}{2}(y_o - \overline{y_o})^T W K_i(\theta) W^T (y_o - \overline{y_o}), i = 1, 2, \ldots, r$   (49)
    where $W = (K_{\epsilon_y} + A_1 K(\theta) A_1^T)^{-1} A_1 \in \mathbb{R}^{d\times N}$ is computed by the SMW formula. The time cost is
    dominated by computing matrix products $(A_1^T K_{\epsilon_y} A_1)$ which takes $O(N)$ linear system solves with the
    noise covariance matrix $K_{\epsilon_y}$ and $O(mN^2)$ time to assemble. Note that $W$ needs to be computed only once
    for all $i$, after which the number of operations depends only on $N$. Therefore the dominant cost is
    independent of $r$ for this step. We only compute and store $W$ in this step.
  Step 3. Draw $N_l$ independent Rademacher vectors $\{u_j\} \in \mathbb{R}^d$. Approximate the second trace term by
      $\alpha_2^i \approx -\frac{1}{2}\mathrm{tr}((K_{\epsilon_y} + A_1 K(\theta) A_1^T)^{-1}(A_1 K_i(\theta) A_1^T))$   (50)
      $\approx -\frac{1}{2N_l}\sum_{j=1}^{N_l} u_j^T W K_i(\theta) A_1^T u_j.$   (51)
  In this step, we only compute $\{A_1^T u_j\}$ and $W^T u_j$ for $j = 1, 2, \cdots, N_l$.The cost is $O(mNN_l)$.
  Step 4. Set $g_i(\theta) = \alpha_1^i + \alpha_2^i$ for $i = 1, 2, \cdots, r$. Assembling (49) and (51) for each $i, j$ takes a total
    $O(rN_lN^2 + mN)$ time.
  Step 5. The left hand side of the score equations is now available as
      $g_i(\theta) = 0,\ i = 1, 2, \cdots, r.$   (52)
    Problem (52) can be solved by, for example, Broyden's method that requires only the left hand side for a
    given $\theta$ and can thus be computed by Step 1-4.
```

In summary, the system of score equations takes $O(N)$ forward solves, $O(N)$ linear system solves with the noise covariance matrix K_{ϵ_y} and an assembly effort of $O(rN^2N_l + mN^2 + mNN_l)$. Each optimization iteration has quasilinear time complexity with n if $N, N_l = log(n)$.

4.2 Joint Process MLE

Given partial observations $y_o \in \mathbb{R}^d$, additional observations of the latent process $z_o \in \mathbb{R}^D$ and based on the covariance model for the joint process (23), the log-likelihood function of the observations (up to an additive constant) is given by

$$\log p(y_o, z_o|\theta) = -\frac{1}{2}\log\det K_{11} - \frac{1}{2}\begin{pmatrix} y_o - \overline{y_o} \\ z_o - \overline{z_o} \end{pmatrix}^T K_{11} \begin{pmatrix} y_o - \overline{y_o} \\ z_o - \overline{z_o} \end{pmatrix}, \tag{53}$$

$$K_{11} = \begin{pmatrix} K_{\epsilon_y} + L_o \mathrm{Cov}(z, z; \theta) L_o^T & L_o \mathrm{Cov}(z, z_o; \theta) \\ \mathrm{Cov}(z_o, z; \theta) L_o^T & K_{\epsilon_z} + \mathrm{Cov}(z_o, z_o; \theta) \end{pmatrix}, \tag{54}$$

The log-likelihood gradient components are given by

$$\begin{aligned} g_i(\theta) =& \frac{1}{2}\begin{pmatrix} y_o - \overline{y_o} \\ z_o - \overline{z_o} \end{pmatrix}^T K_{11}^{-1} \begin{pmatrix} L_o \mathrm{Cov}_i(z,z;\theta) L_o^T & L_o \mathrm{Cov}_i(z,z_o;\theta) \\ \mathrm{Cov}_i(z_o,z;\theta) L_o^T & \mathrm{Cov}_i(z_o,z_o;\theta) \end{pmatrix} K_{11}^{-1} \begin{pmatrix} y_o - \overline{y_o} \\ z_o - \overline{z_o} \end{pmatrix} \\ &- \frac{1}{2}\mathrm{tr}\left(K_{11}^{-1} \begin{pmatrix} L_o \mathrm{Cov}_i(z,z;\theta) L_o^T & L_o \mathrm{Cov}_i(z,z_o;\theta) \\ \mathrm{Cov}_i(z_o,z;\theta) L_o^T & \mathrm{Cov}_i(z_o,z_o;\theta) \end{pmatrix} \right), \end{aligned} \tag{55}$$

subsequently, the score equations are defined by $g_i(\theta) = 0$, $i = 1, 2, \ldots, r$.

As in the rest of the paper, we efficiently approximate the components $g_i(\theta)$ by means of low-rank approximations of $\mathrm{Cov}(z, z_o; \theta)$ and $\mathrm{Cov}(z_o, z_o; \theta)$, as well as their derivatives with respect to θ_i (denoted here with the subscript i) by interpolation. A scalable approach for solving joint process MLE problem by score equations is summarized in Algorithm 5.

In summary, computing the gradient components for the score equations takes $O(N)$ forward solves, $O(N)$ linear system solves with the noise covariance matrix, and $O(rN_lN^2 + (m+n)NN_l + (m+n)N^2)$ operations for the assembly of the various matrices. For any θ, computing these gradient components has a quasilinear time complexity with n if $N, N_l = log(n)$.

4.3 Estimation of Fisher Information Matrix

The expected Fisher information matrix is commonly used in providing confidence intervals on the estimates of θ by means of asymptotic analysis. Specifically, denote the solution of the score equations by $\hat{\theta}$ and the Fisher information matrix by $\mathcal{I}(\hat{\theta})$. Then, as the sample size goes to infinity, one expects that

$$(\hat{\theta} - \theta_{\mathrm{true}}) \xrightarrow{d} \mathcal{N}(0, \mathcal{I}(\hat{\theta})^{-1}). \tag{60}$$

Therefore, for each component of θ, the confidence interval can be approximated by

$$\hat{\theta}_i \pm c\sqrt{(\mathcal{I}(\hat{\theta})^{-1})_{ii}}, \tag{61}$$

where c is the appropriate critical value. Furthermore, for multivariate Gaussian distribution where the uncertainties only occur in the covariance, each entry in the expected Fisher information matrix is given by [30]

$$\mathcal{I}_{i,j}(\theta) = \frac{1}{2}\mathrm{tr}(\Sigma(\theta)^{-1}\Sigma_i(\theta)\Sigma(\theta)^{-1}\Sigma_j(\theta)), \tag{62}$$

where $\Sigma(\theta)$ denotes the autocovariance matrix in latent or joint process MLE and $\Sigma_i(\theta) = \frac{\partial}{\partial \theta_i}\Sigma(\theta)$. The trace term can be again approximated by the stochastic trace estimator

$$\hat{\mathcal{I}}_{i,j}(\theta) \approx \frac{1}{2N_l}\sum_{l=1}^{N_l} u_l^T \Sigma(\theta)^{-1}\Sigma_i(\theta)\Sigma(\theta)^{-1}\Sigma_j(\theta) u_l. \tag{63}$$

where u_l are N_l independent Rademacher random vectors $u_l \in \mathbb{R}^d, l = 1, 2, \ldots, N_l$. The covariance components in (63) are computed by the same low-rank approximations as in the rest of the paper. We recall from Algorithm 4 that $A_1 = L_o C_z^T$ is computed when setting up the score equations. From (63) it follows that

$$\begin{aligned} \hat{\mathcal{I}}_{i,j}(\theta) \approx& \frac{1}{2N_l}\sum_{l=1}^{N_l} u_l^T (K_{\epsilon_y} + L_o\mathrm{Cov}(z,z;\theta)L_o^T)^{-1}(L_o\mathrm{Cov}_i(z,z;\theta)L_o^T)(K_{\epsilon_y} + L_o\mathrm{Cov}(z,z;\theta)L_o^T)^{-1} \\ & (L_o\mathrm{Cov}_j(z,z;\theta)L_o^T)u_l. \\ \approx& \frac{1}{2N_l}\sum_{l=1}^{N_l} u_l^T (K_{\epsilon_y} + A_1K(\theta)A_1^T)^{-1}(A_1K_i(\theta)A_1^T)(K_{\epsilon_y} + A_1K(\theta)A_1^T)^{-1}(A_1K_j(\theta)A_1^T)u_l \end{aligned} \tag{64}$$

Algorithm 5: Joint Process Score Equations

Step 1. Compute $A_1 = L_o C_z^T \in \mathbb{R}^{d\times N}$, $A_2 = L_p C_z^T \in \mathbb{R}^{(m-d)\times N}$ by $O(N)$ forward solves.

Step 2. Compute the approximation of the first term in score equations,

$$\alpha_1^i \approx \frac{1}{2}\begin{pmatrix} y_o - \overline{y_o} \\ z_o - \overline{z_o}\end{pmatrix}^T \left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K(\theta) & K(\theta) \\ K(\theta) & K(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)^{-1}\begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}$$
$$\begin{pmatrix} K_i(\theta) & K_i(\theta) \\ K_i(\theta) & K_i(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K(\theta) & K(\theta) \\ K(\theta) & K(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)^{-1}$$
$$\begin{pmatrix} y_o - \overline{y_o} \\ z_o - \overline{z_o}\end{pmatrix}. \tag{56}$$

Notice that the expression is symmetric. We first compute the term,

$$W = \left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K(\theta) & K(\theta) \\ K(\theta) & K(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)^{-1}\begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix} \tag{57}$$

The time cost is dominated by computing the matrix products $(A_1^T K_{\epsilon_y}^{-1} A_1)$ and $(C_{z_o} K_{\epsilon_z}^{-1} C_{z_o}^T)$ which require $O(N)$ linear system solves with observation noise covariance matrices and $O((n+m)N^2)$ assembly time. This needs only to be done once for all $i = 1, 2, \ldots, r$. We will store W and assemble the entire expression (56) later on.

Step 3. Draw N_l independent Rademacher vector $\{u_j\} \in \mathbb{R}^{d+D}$. Approximate the trace term in (55) by

$$\alpha_2^i \approx -\frac{1}{2N_l}\sum_{j=1}^{N_l} u_j^T W \begin{pmatrix} K_i(\theta) & K_i(\theta) \\ K_i(\theta) & K_i(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix} u_j. \tag{58}$$

Same as in Algorithms 4 in this step we only compute the terms $(W^T u_j)$ and $\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix} u_j$ for each j. Store these vectors and assemble the whoe term later on. This step requires $O((n+m)NN_l)$ time to compute.

Step 4. We now set $g_i(\theta) = \alpha_1^i + \alpha_2^i$ for $i = 1, 2, \cdots, r$. The main computational expense consists of computing (56) and (58) for each i, j, which takes $O(rN_l N^2 + (n+m)N)$ time.

Step 5. The left hand side of the score equations is now available as

$$g_i(\theta) = 0, i = 1, 2, \cdots, r. \tag{59}$$

The problem can be subsequently solved by using, for example, Broyden's method and *Step 1-4* to compute the components of the gradient.

Observing the structure of the term, we will evaluate the Fisher information matrix in the following sequence. First we compute $W = (K_{\epsilon_y} + A_1 K(\theta) A_1^T)^{-1} A_1$ by SMW formula. Note that the W component for SMW only needs to be assembled once. This step takes $O(N)$ K_{ϵ_y} linear system solves and $O(mN^2)$ time. Then forming the $N \times N$ matrix $U = A_1^T W$ takes $O(mN^2)$ time. After that we compute the right terms $\{A_1^T u_l\}$, $l = 1, 2, \cdots, N_l$ with $O(mNN_l)$ cost and the left terms $\{w_l\}$ where $w_l = W^T u_l$, $l = 1, 2, \cdots, N_l$ with $O(mNN_l)$ cost. Now we finish our precomputing process with a cost independent from r. Then we assemble the entries in the Fisher information matrix. For each i, j, we compute N_l terms to approximate the trace. Each term takes form $w_l^T K_i(\theta) U K_j(\theta) A_1^T u_l$ with $O(N^2)$ cost. Since we have r^2 entries in Fisher information matrix, the total assembling cost is $O(r^2 N_l N^2)$. To summarize, the whole Fisher information matrix takes $O(N)$ K_{ϵ_y} linear system solves and an assembly cost of

$O(mN(N+N_l)+r^2N_lN^2)$ time to compute. The cost of inverting the expected Fisher information matrix can be ignored.

Similarly, for computing the Fischer matrix in the joint process model, we draw N_l independent Rademacher vector $\{v_l\} \in \mathbb{R}^{d+D}$, $l = 1,2,\ldots,N_l$. Using the low-rank approach we get the following approximation for its entries.

$$\begin{aligned}\hat{\mathcal{I}}_{i,j}(\theta) \approx & \frac{1}{2N_l}\sum_{l=1}^{N_l} v_l^T \left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K(\theta) & K(\theta) \\ K(\theta) & K(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)^{-1}\begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix} \\ & \begin{pmatrix} K_i(\theta) & K_i(\theta) \\ K_i(\theta) & K_i(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K(\theta) & K(\theta) \\ K(\theta) & K(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)^{-1} \\ & \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K_j(\theta) & K_j(\theta) \\ K_j(\theta) & K_j(\theta)\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix} v_l \end{aligned} \tag{65}$$

The computational cost analysis is very similar as the one for the Fischer matrix of the latent process model (64). The precomputing step takes $O(N)$ linear system solves with the observation noise covariance matrix and $O((n+m)N(N+N_l))$ vector and matrix assembly time. Assembling the Fisher information matrix takes $O(r^2N_lN^2)$ time. The cost of inverting the expected Fisher information matrix can be ignored.

Since the dimension of the parameters is finite, taking $N, N_l = O(\log n)$ provides us a scalable approach of estimating the Fisher information matrix for estimating the parameters of our uncertainty model.

5. NUMERICAL EXPERIMENTS

In this section, we present numerical tests based on both synthetic and realistic datasets. The synthetic case is based on the 1D viscous Burger's equation and the corresponding model-generated data. The realistic case concerns reanalyzed satellite-observed water vapor and wind velocity data. For the synthetic case, we investigated all model and computation features described in this paper, including numerical accuracy of parameter estimations and uncertainty quantification when carrying out low-rank-based reduction, §2.2, comparisons of the different covariance models §3.1 and §3.2, effect of higher-order terms §3.3, accuracy of the confidence interval estimates when carrying out maximum likelihood §4.3, and the measured computational scaling of the approximated algorithm. For the real atmospheric dataset, we consider only linear models and we focus primarily on improvement in kriging accuracy compared to the case when the cross-covariance information is ignored and model comparison between §3.1 and §3.2. Since the contribution of this work is in providing a scalable approach for both the kriging and the maximum likelihood tasks for implicitly-defined Gaussian process models, we will be less interested in doing an extensive modeling comparison. We will focus on demonstrating that the approximating nature of our method does not significantly alter the predictive power of a physics-based approach to a modeling of the covariance, which was by itself demonstrated in [25].

5.1 One-dimensional Viscous Burger's Equation

Burger's equation is a fundamental partial differential equation which combines the nonlinear advection and the linear diffusion. It can be investigated in $1+1$ (one spatial dimension and one temporal dimension) dimensional settings and is a standard test case for data assimilation algorithms [31–35]. The equation takes the following form:

$$\frac{\partial w}{\partial t} + \frac{1}{2}\frac{\partial (w^2)}{\partial x} = \nu \frac{\partial^2 w}{\partial x^2} + f,\ (x,t) \in (0,1)\times(0,T). \tag{66}$$

For assessing the effectiveness of our uncertainty quantification approach we include an additional external forcing term f which is deterministic and time-independent, we select a smaller diffusion coefficient, and choose more complicated initial conditions to engender more long-lasting chaotic behavior.

To numerically solve the equation, denote the value of its approximation at grid point $(j\Delta x, m\Delta t)$ by w_j^m and of the constant forcing as f_j. We construct a forward solver of the system using the following finite difference scheme:

$$\frac{w_j^{m+1} - w_j^m}{\Delta t} + \frac{(w_{j+1}^m)^2 - (w_{j-1}^m)^2}{4\Delta x} - \frac{\nu}{(\Delta x)^2}(w_{j+1}^{m+1} - 2w_j^{m+1} + w_{j-1}^{m+1}) - f_j = 0. \tag{67}$$

To apply our framework to this problem, we find the solution of the problem on the two dimensional grid given the initial condition I and boundary condition B. Write $z^T = (B^T, I^T)$. Then we define $w = G(z)$ where w is the solution produced by the numerical scheme (67). In all our synthetic experiments we take y_o to consist of a subset of the components of w to which we add Gaussian observation noise $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ and y_p to consist of another such subset. The mappings F_o, F_p required by our framework (16)-(17) will be obtained from the corresponding components of G. The actual observation and prediction components y_o, y_p may change to suit the validation we try to carry out and we will define them at multiple points in the subsequent material below.

To generate the synthetic data we use the following settings:

$$\nu = 0.01,\ T = 0.1,\ f = \pi \sin(\pi x)(\cos(\pi x) + \nu\pi), \tag{68}$$

$$I \sim \mathcal{N}(sin(\pi x), \alpha_I \exp(-\frac{r^2}{2l^2})), \tag{69}$$

$$B \sim \mathcal{N}(0, \alpha_B \exp(-\frac{r^2}{2(lT)^2})), \tag{70}$$

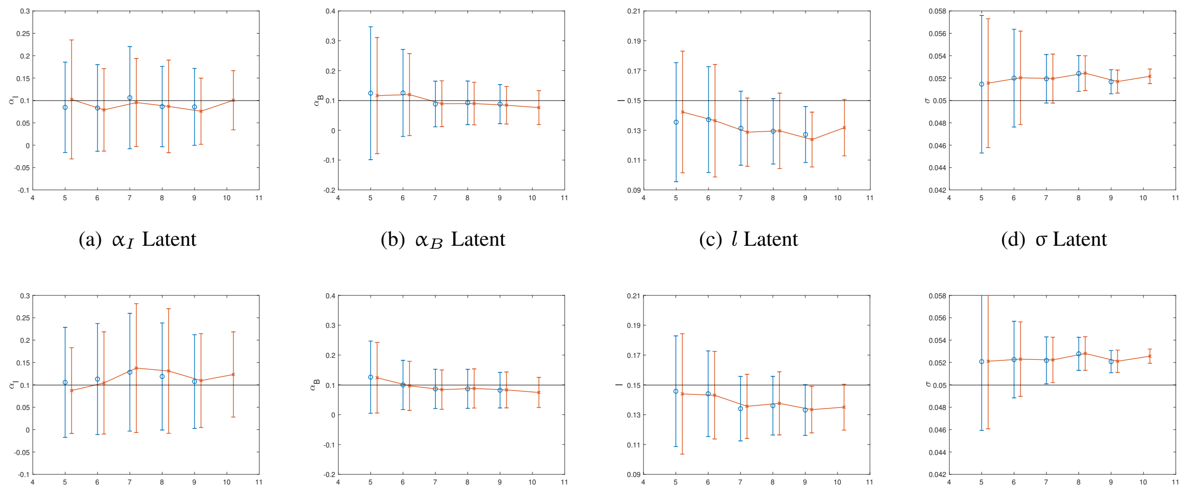
$$\alpha_I = 0.1,\ \alpha_B = 0.1,\ l = 0.15,\ \sigma = 0.05. \tag{71}$$

To simplify the notation, we describe the normal processes defining the distributions of I and B in terms of the mean functions and covariance kernels, the distribution used in computations is obtained by sampling them at the chosen mesh points. In the definition of the covariance kernels r is the distance between two points and l is the length scale parameter. We may vary the mesh size Δx, Δt but we always maintain $\frac{1}{\Delta x} = \frac{T}{\Delta t}$ so that we have equal number of mesh points in the two dimensions. Denote the number of mesh points $\frac{1}{\Delta x} = \frac{T}{\Delta t} = k$ then $B \in \mathbb{R}^{2k}$, $I \in \mathbb{R}^k$. The dimension of w in (67), and, thus the potential dimension of y would be $O(k^2)$. For a realistic setup the number of prediction points should be of an order at most comparable with the data size ; we will thus choose the dimension of y_o and y_p to be $O(k)$. The size of the GP regression problem will thus be growing at most linearly with k.

When carrying out our low rank approximation, §2.2, we use $N = 12 \log k$ interpolation points to approximate $\mathrm{Cov}(z, z)$. Also we use $N_l = 20 \log k$ random vectors for stochastic trace estimation when approximating trace terms in §3.3, §4.

5.1.1 Numerical Accuracy of Parameter Estimation

First we explore the numerical accuracy of estimating the covariance model parameters by our scalable maximum likelihood approach from §4, inclusive of the computation of the confidence intervals using expected Fisher information matrix from §4.3. The parameters here are $\theta = (\alpha_I, \alpha_B, l, \sigma)$, with their true values given by (71). The data is created by simulating the PDE on $k = 2^{11}$ mesh points in both spatial and temporal dimensions which result in a total number of 2^{22} data point in regular grid. Then all subsequent calculations for obtaining the maximum likelihood estimates and their confidence intervals, including the ones for getting the functions G and F that require the discretization (67), will be carried out on $k = 2^q$ mesh points in each dimension for q ranging from 5 to 10. This mimics the real situation where the function F is "black box" but allows the introduction of an external mesh parameter. For each k, we randomly sample $d = 10k$ observations of the output process for latent process MLE §4.1 and additional $D = 0.6k$ observations of the latent process for joint process MLE §4.2. We note that a random protocol was also used in [25]. In our experiments we solve the nonlinear score equations (52) and (59) by the Matlab `fsolve` function (which uses divided differences approximations of the gradient) with the `trust-region-dogleg` algorithm. The initial guess is the same as the truth and the stopping condition is set to be relative tolerance of 10^{-8}. Finding a good initial point is always difficult in maximum likelihood calculations [30] and in this sense choosing the real parameters



(a) α_I Latent (b) α_B Latent (c) l Latent (d) σ Latent

(e) α_I Joint (f) α_B Joint (g) l Joint (h) σ Joint

FIG. 1: Parameter estimations and corresponding 95% confidence intervals computed via expected Fisher information matrix. We solve the nonlinear system of score equations (45),(55) using the physics-based covariance model with number of mesh points $k = 2^q$ in each dimension for q ranging from 5 to 10. Red lines with crosses represent MLE using low-rank approximations. The first row describes the results for latent process MLE §4.1. The second describes the results for joint process MLE §4.2. MLE results using exact score equations are provided for $q = 5$ to 9 using blue lines with circles. For $q = 10$ the exact MLE calculation ran out of memory.

is somewhat unsatisfactory. Note, however, that here we are mostly concerned with the ability of the approximated likelihood calculations that we describe in §4 to result in accurate parameter estimation rather than characterizing the difficulty of dealing with the nonlinearity of the likelihood proper. We thus find this initialization approach to be acceptable given our goal.

The estimated parameters and corresponding 95% confidence intervals are summarized in Figure 1 along with the corresponding estimates using the exact likelihood score equations for both latent model (45) and joint model (55). We note that the *exact* calculation is exact for the Gaussian process model obtained by the *linearization* of F, which is only an approximation of the true likelihood, which is most likely not Gaussian. As we can observe, the parameter estimates by our approximated approach and exact log-likelihood are fairly close, both in terms of the estimated parameters values and the width of confidence intervals. On the other hand, at large q and k, the exact parameters may be outside the confidence interval, though obviously so only for σ. At that point the error from the linearization exceeds the statistical error. One way to mitigate this (though, also approximate) is to use the nonlinear corrections from §3.3 for the likelihood too. It has, however, proven difficult to extend the algebra from §3.3 for using the derivative of the covariance as well at this time, and we will pursue that direction in future work. Based on the results on Figure 1 we note that the approximate likelihood calculations produce results very close to the exact likelihood, even if both models are incorrect due to the nonlinearity. Moreover, as we show in the subsequent sections, the parameter estimates we produce with this approximate (and biased) likelihood calculation do improve the GP regression estimates when plugged into the covariance expression. Finally, note that the parameter error is within 5% in any case.

5.1.2 Comparison of Covariance Models

We now examine the predictive performance of the physics-based covariance model §2, inclusive of the low-rank §2.2 and implicit formulation approximation §2.3 effects. We compare the latent process kriging §3.1 and joint process kriging §3.2 with independent process kriging which ignores the relations between the two processes by removing the cross-covariance. The latter setup aims to exhibit the features of the most prevalent approaches in modeling where a difficulty of expressing valid cross-covariances models result in the need for modeling different observables independently.

Independent kriging. In this setting we separate the output and latent processes by considering their covariances independently and setting the cross-covariance be zero. We use the following covariance structure:

$$K_{11} = \begin{pmatrix} \mathrm{Cov}(y_o, y_o) + \sigma^2 I_d & 0 \\ 0 & \mathrm{Cov}(z_o, z_o) + \sigma^2 I_D \end{pmatrix}, \tag{72}$$

$$K_{21} = \begin{pmatrix} \mathrm{Cov}(y_p, y_o) & 0 \\ 0 & \mathrm{Cov}(z_p, z_o) \end{pmatrix}, \tag{73}$$

$$\mathcal{M}_{\mathrm{ind}} = \begin{pmatrix} m(y_p) \\ m(z_p) \end{pmatrix} + (K_{21})(K_{11})^{-1} \begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o) \end{pmatrix}. \tag{74}$$

At this stage the ideal comparison would be with a well-chosen Gaussian kernel to model $\mathrm{Cov}(y_o, y_o)$ and $\mathrm{Cov}(z_p, z_o)$ separately. Since our model originates in the approximation of a nonlinear equation, the implied covariance of the linearized process for meaningful boundary conditions will not be stationary (since the advection would depend on space). Modeling a nonstationary process in the output space is a difficult endeavor with no simple approach (other than using a simple stationary kernel class anyways, e.g squared exponential [1]). Since our focus is to understand the effect of ignoring covariance and to get access to a reasonable and easy to set up model we still use the physics-based covariance model $\mathrm{Cov}(y_o, y_o) = L_o \mathrm{Cov}(z, z) L_o^T$ and $\mathrm{Cov}(y_p, y_o) = L_p \mathrm{Cov}(z, z) L_o^T$, where L_o, L_p correspond to the Jacobians of mapping F_o, F_p, respectively. Thus, we consider the effect of z on y for the purpose of the autocovariance but not for the purpose of the crosscovariance. While it would be more principled to set up a model which completely ignore the physics, we believe that using the "best model" for the autocovariance in this circumstance would allow us to sharply assess the effects of ignoring the cross-covariance, an approach also used in the paper that inspired this work [25]. Moreover, this does not affect the main concern of our paper, whether the approximation we use to make the computation scalable still presents enough accuracy overall.

We note that, if we make this choice, the prediction carried out by (74) would be identical to the latent process approach §3.1, provided that the parameters θ stay the same. Since, however, in the case of the model (72) – (73) we use both z_o and y_o data to estimate θ, whereas for the latent process model in §3.1 we use only y_o, the parameters θ and thus, the kriging results, will be different. The joint process approach from §3.2, however, will use values of both y_o and z_o for both estimating θ as well as for kriging.

Finally we consider model using the joint process covariance model structure (23)-(25) with the hyperparameters defined above (71) that were used to produce the data (the "true" parameters). We denote this true model by $\mathcal{M}_*$.

We then take a fixed mesh size with $k = 200$, which results in totally 40000 data points for the numerical approximation of PDE using (67). We randomly pick $d = 400$ data points in the output field w as observations y_o, which corresponds to 1% of the total field to observe. In the joint process case §3.2, we have $D = 20$ additional observations each from the initial conditions and boundary conditions as latent observations z_o which are 10% of the latent variables. Note that the dimension of w is roughly the square of the dimension of z; so the dimension of the observations of z and w are comparable. Note also that the setting is similar to [25].

Since the computational complexity is not a major concern in this subsection, we predict the entire remaining field $y_p = w \setminus y_o$ and $z_p = z \setminus z_o$. We first generate one sample using the hyperparameters (71) termed calibration sample, which is used to fit the covariance models $\mathcal{M}_1$ (18), $\mathcal{M}_2$ (23)-(25),$\mathcal{M}_{\mathrm{ind}}$ (72)-(74) as described in §5.1.1. Then we further generate 50 independent samples used for tests in our experiment, termed validation samples. The results are averaged and summarized in the upper block of Table 1.

We note that the RMS error on y_p of the three physics-based covariance models $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_*$ is significantly smaller (by factors of at least 6) on the validation samples than the error of the independent kriging $\mathcal{M}_{\mathrm{ind}}$, which

TABLE 1: Predictive RMS error (excluding observed locations). The observation density is 1% for w and 10% for z, though the dimension of the observation vectors is comparable. The observations are chosen randomly in the field. The predictions are conducted for all the remaining field in w and z except the observed positions. The first four rows contains covariance models without higher-order terms from §5.1.2 and the last three rows contains covariance models with higher-order terms from §5.1.3.

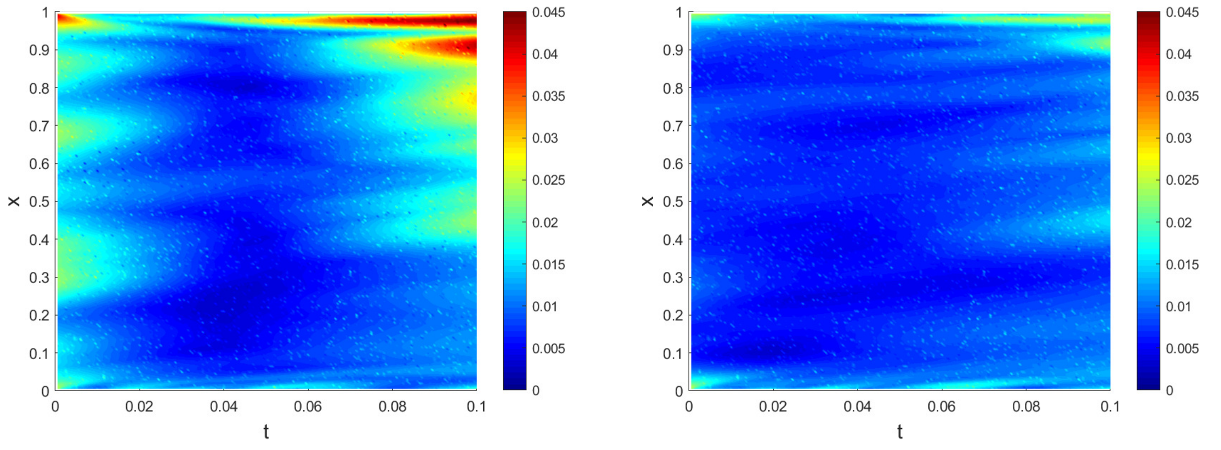
	Validation Samples		Calibration Sample	
Models	y_p	z_p	y_p	z_p
$\mathcal{M}_1$	0.0241	-	0.0296	-
$\mathcal{M}_2$	0.0211	0.0401	0.0195	0.0495
$\mathcal{M}_{\text{ind}}$	0.1422	0.4780	0.1189	0.5107
$\mathcal{M}_*$	0.0211	0.0339	0.0195	0.0514
$\mathcal{M}_1^+$	0.0228	-	0.0247	-
$\mathcal{M}_2^+$	0.0180	0.0334	0.0175	0.0325
$\mathcal{M}_*^+$	0.0207	0.0331	0.0117	0.0335

demonstrates the contribution of the cross-covariance terms to the predictions. Furthermore, the two models using joint process kriging $\mathcal{M}_2$ and $\mathcal{M}_*$ are of similar performance and are both better by about 10% on the validation samples compared to the latent process kriging $\mathcal{M}_1$ due to the additional information conveyed by the observations z_o. This improved accuracy occurs despite the fact that we have low-rank §2.2, differentiation §2.3, and linearization approximation errors in the computation of the covariance. While our starting the MLE at the true parameters allows for the possibility of other local minima, it is clear that the approximated likelihood has minima that are mostly indistinguishable from the true parameters for the purpose of the accuracy of the kriging.

5.1.3 Correction of Nonlinearity

Since the Burger's equation is nonlinear, we now consider adding higher-order terms in the covariance model to account for the nonlinear behaviors generated by the equation using the methods described in §3.3. To demonstrate the benefit gained from the higher-order terms, we consider the same experiment settings as in Section 5.1.2. Note that we only consider the correction of nonlinearity on models $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_*$. The models with higher-order terms included are denoted by $\mathcal{M}_1^+$, $\mathcal{M}_2^+$, $\mathcal{M}_*^+$, respectively. Specifically, $\mathcal{M}_1^+$ follows the latent process kriging with higher-order terms Algorithm 3. $\mathcal{M}_2^+$, $\mathcal{M}_*^+$ follow the joint process kriging with higher-order terms Algorithm 6. When setting the parameters $\theta = (\alpha_I, \alpha_B, l, \sigma)$ in $\mathcal{M}_1^+$ and $\mathcal{M}_2^+$ we use the parameters fitted by the approximated MLE described in §4 but for the linear approximation process only; that is the same process described in §5.1.1 (in other words we consider the nonlinearity when kriging, but not when carrying out MLE) whereas for $\mathcal{M}_*^+$ we use the true parameters (71). The performance of the nonlinearly corrected models on the same calibration sample and validation samples is summarized in the lower block of Table 1. We note that the performance of all three models is better, albeit marginally in most cases (in the $2\% - -14\%$ range), than their counterparts using covariance models without higher-order terms $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_*$ because the higher-order terms help to account for the nonlinearities. The improvement is the largest in the joint model $\mathcal{M}_2$ to the point where it exceeds the one with the true parameters. It is unlikely that this will hold for all possible experimental validation setups, but the improvement if RMSE is clear across all models when considering nonlinear effects even if small.

Additionally, to provide a visual comparison of the predictions, we combine the predicted data and the observations to reconstruct the whole field w. To demonstrate the effectiveness of the higher-order terms, we compute the average error distribution over the 50 validation samples for joint process kriging models $\mathcal{M}_2$ and $\mathcal{M}_2^+$. The comparison is shown in Figure 2. As we can observe, even if the reduction in RMSE is not large, the patern of the error is much improved. The error is uniformly smaller for covariance models with higher-order terms (which indicates that in the max norm the error improvement is significant) which demonstrates the usefulness of the nonlinear correction. To support this observation, we report quartiles to describe the distribution of the error surface. The three quartiles



(a) Without higher-order terms (b) With higher-order terms

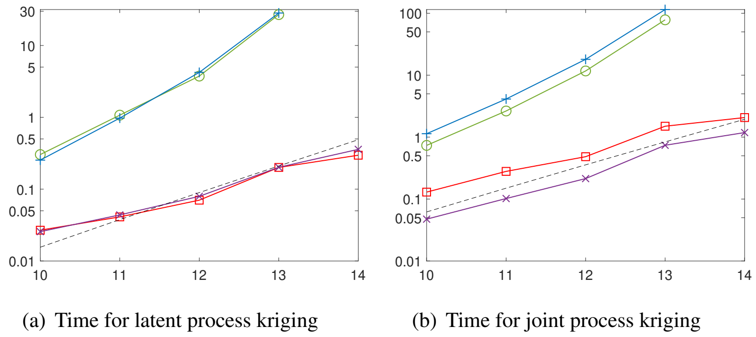
FIG. 2: Error distribution of the predicted field compared with the true data §5.1.3. **Left**: Results from joint process kriging $\mathcal{M}_2$. The covariance model follows (23)-(25) which includes up to second-order terms corresponding to (4). **Right**: Results from joint process kriging with higher-order terms $\mathcal{M}_2^+$. The covariance model follows (30)-(32) which includes up to fourth-order terms corresponding to (5)(6). The higher-order terms should account for part of the nonlinear behavior of Burger's equation.

(Q_1, Q_2, Q_3) of the error are 0.078, 0.0109, 0.0153 for model $\mathcal{M}_2$ in Figure 2 (a). In contrast, the three quartiles for model $\mathcal{M}_2^+$ in Figure 2 (b) are 0.067, 0.082, 0.0102. Therefore the spatial distribution of the error has much thinner tail for model $\mathcal{M}_2^+$ compared to model $\mathcal{M}_2$, indeed the third quartile of the error for $\mathcal{M}_2^+$ is smaller by about a third compared to the third quartile of the error for $\mathcal{M}_2$.

5.1.4 Demonstrating Quasilinear Scaling

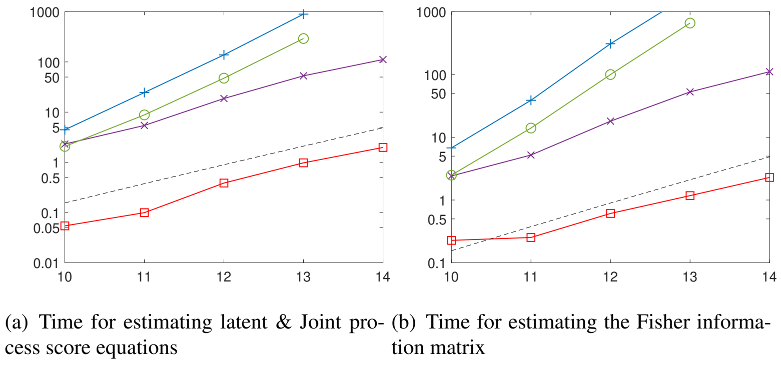
We now present numerical experiments to demonstrate the scalability of our proposed approach when computing the covariance parameters θ by the maximum likelihood approach in the §5.1.1 setting, by carrying out the computation of the approximated score equations §4, followed by the kriging procedures described in §3 to predict the field of interest at prescribed points. To vary the size of the problem we take mesh size $k = 2^q$ for q ranging from 10 to 14. We randomly observe $d = 3k$ data points from the output process w and $D = 1.5k$ data points from the latent process z. Then we predict $3k$ points for the output process and $1.5k$ points for the latent process at sites that are also randomly chosen. The random part of our experiment is repeated 50 times. For both latent process kriging §3.1 and joint process kriging §3.2, we compare the average time cost over the 50 of our workflow without higher-order terms (Algorithm 1&2) and with higher-order terms (Algorithm 3&6). For the latent process MLE §4.1 and joint process MLE §4.2, we compare the time of solving the approximated score equations (45) (55) with the time of solving the exact ones (where the covariance matrices are exactly computed and not approximated by interpolation); the timings reported are for the computation on the first sample only. The results are summarized in Figure 3 and 4. Lines corresponding to the theoretical scaling $O(k \log^2 k)$ are added to each plot to demonstrate the quasilinear scaling of our approaches.

As we can observe, the scaling of proposed approximated approaches in latent and joint process kriging follow the $O(k \log^2 k)$ theoretical line. Comparing the algorithm with and without higher-order terms, we observe that the algorithm without higher-order terms scales slightly better due to the extra operations when considering higher-order terms. This effect will be alleviated as datasets grow larger. For the score equations computation and Fisher information matrix estimation, the quasilinear scaling is clearly demonstrated in Figure 4. Note that our approximated approach requires stochastic estimation of the trace term, which is not as efficient as direct computation in small datasets, i.e. $q = 10$ in Figure 4.



(a) Time for latent process kriging (b) Time for joint process kriging

FIG. 3: Times taken (in seconds) to solve the GP regression problem by (a) latent process kriging (Algorithm 1&3) , (b) joint process kriging (Algorithm 2&6). We present the time to compute latent/joint process kriging without higher-order terms exactly (circle) and approximately (square), latent/joint process kriging with higher-order terms exactly (plus) and approximately (cross). All results are averaged over 50 tests. Note that the y-axis is in logarithmic scale. The lines corresponding to theoretical $O(k\log^2 k)$ scaling (dash lines) are added to each plot.



(a) Time for estimating latent & Joint process score equations (b) Time for estimating the Fisher information matrix

FIG. 4: Times taken (in seconds) to evaluate (a) latent & Joint process score equations (Algorithm 4&5) and (b) the expected Fisher information matrix via equation (64), (65). For both plots, we present the time to evaluate the score equations or the expected Fisher information matrix for latent process exactly (circle) and approximately (square) and for joint process exactly (plus) and approximately (cross).All results are averaged over 50 tests. Note that the y-axis is in logarithmic scale. The lines corresponding to theoretical $O(k\log^2 k)$ scaling (dash lines) are added to each plot.

In summary for the synthetic data case, we conclude that our approach scales quasilinearily §5.1.4, that the approximations that we have proposed in order achieve still allow an accurate recovery of the parameters defining the shape of the uncertainty §5.1.1, that the accurate treatment of the cross-covariance significantly improves prediction §5.1.2, and that the including the nonlinear corrections to the covariance slightly improves the RMSE of the prediction, but may significantly help with removing complex error artifacts §5.1.3.

5.2 Real Data Experiment

In this subsection, we consider realistic climate observation data and a simple tropical climate model connecting key quantities as the physical information we will use to inform our covariance model. The model equation is given by

$$\frac{\partial w}{\partial t} = -\tilde{Q}\nabla u - \frac{1}{\tau_w} w + b_w \nabla^2 w + D_w \dot{W}, \tag{75}$$

where u is the first baroclinic mode of the zonal wind velocity anomalies , w is the water vapor anomalies in middle troposphere and $\dot{W}$ is Gaussian white noise. The core dynamic (75) uses a modified equation from [36]. The term

$-\tilde{Q}\nabla u$ is a traditional model of convective adjustment of water vapor caused by wind. The term $-\frac{1}{\tau_w}w$ accounts for moisture sink associated with deep convection and precipitation. $b_w\nabla^2 w$ represents the diffusion of water vapor and $D_w\dot{W}$ represents stochastic forcing. In short, this model represents the effect of precipitation, natural diffusion and turbulent advection in a simplified, linearized form. Besides the deterministic dynamic components, the stochastic moisture forcing is, in part, representative of mesoscale convective processes that are not represented by the larger-scale dynamics in (75). Therefore it is reasonable to assume that such processes are approximately spatio-temporally uncorrelated. The parameter values $\tilde{Q} = 0.45$, $\tau_w = 3.99$, $b_w = 0.7$, $D_w = 0.003$ are all dimensionless and are suggested by [36].

To numerically solve the model (75), we use the initial condition w_0 and random field u as model inputs with periodic boundary condition. The equation can be solved using Euler-Maruyama method [37] and also in Fourier space [38]. We recommend to solve in Fourier modes since a semi-analytical solution [38] exists for each Fourier mode which can significantly reduce the time cost of numerical simulation.

To apply our framework, we numerically solve w as output process and we treat the concatenation of the initial condition w_0, the random process u and $\dot{W}$ as the latent process of our framework, i.e. $z^T = (w_0^T, u^T)$ in our framework §(16)-(17). Similar to §5.1, we define $w = G(z)$ where w is the solution produced by semi-analytical solver of stochastic differential equation (75). In the following test, we denote y_o as the observed subset of w which also includes additional observation noise and y_p as the another subset which is going to be predicted. The mapping F_o, F_p required by our setup (16)-(17) are obtained by corresponding components of mapping G.

5.2.1 Data Source and Data Processing

To associate the model (75) with suitable real dataset, we argue that the zonal wind velocity serves as a natural surrogate of u and the relative humidity (mixing ratio) in 500 hPa pressure level is a common surrogate of the water vapor w in middle troposphere.

The data source we used here is the National Centers for Environmental Prediction–National Center for Atmospheric Research (NCEP-NCAR) reanalysis project [39]. The daily zonal wind velocity and relative humidity data are used. Both datasets have a spatial resolution of 2.5° × 2.5° and a daily temporal resolution from 1 January 1979 to 31 December 2011. The connections between model variables and observations are largely based on previous work [40–42].

The influence of seasonal cycle has been removed from both datasets and a further 120-day-mean of previous 120-day signal is subtracted, as is recommended by the Climate Variability and Predictability [43]. Then both datasets are projected to the parabolic cylinder functions and only the first meridional mode is used. This step converted the two spatial dimensional data to one dimensional which facilitates our computation. Then the datasets have two dimensions remaining (one spatial dimension and one temporal dimension). Furthermore, we treat the observation data as true signal and generate artificial observations by adding independent Gaussian noise $\epsilon_w \sim \mathcal{N}(0, \sigma_w^2 I)$ and $\epsilon_u \sim \mathcal{N}(0, \sigma_u^2 I)$ to water vapor data w and zonal wind velocity data u respectively. Parameters σ_w and σ_u are taken to be 15% of the standard deviation of each true signal similar to other previous studies e.g. [44].

Note that the real data are normalized to have the same variance as the model data before using. This differs from [36] where the data are nondimensionalized using the standard equatorial reference scale [41] because here we use anomalies in our modified model equation. For more detailed raw data preprocessing methods please refer to [41]. In Figure 5 we show the observational relative humidity data for all longitudes from Dec 1982 to Apr 1983.

5.2.2 End-to-end Numerical Tests

We combine our proposed scalable MLE parameter estimation and GP regression together and test their performance. Four different covariance structures are compared here. The latent process kriging $\mathcal{M}_1$ (18) and the joint process kriging $\mathcal{M}_2$ (23)-(25) follow the same design as we discussed in §3. Additionally the joint independent kriging $\mathcal{M}_{\text{ind}}$ ignores the cross-covariance between the latent process z and the output w and is formulated as follows:

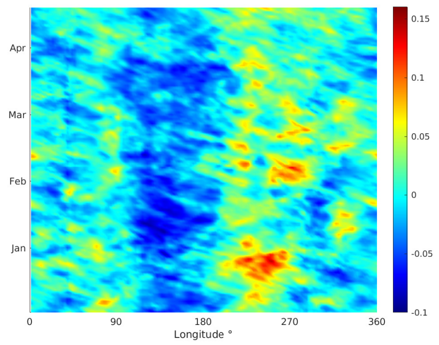


FIG. 5: Observations of relative humidity at 500 hPa pressure level. The time period is from Dec 1 1982 to Apr 18 1983. The data have been nondimensionalized.

$$K_{11} = \begin{pmatrix} \mathrm{Cov}(y_o, y_o) + \sigma_w^2 I & 0 \\ 0 & \mathrm{Cov}(z_o, z_o) + K_{\epsilon_z} \end{pmatrix}, \tag{76}$$

$$K_{21} = \begin{pmatrix} \mathrm{Cov}(y_p, y_o) & 0 \\ 0 & \mathrm{Cov}(z_p, z_o) \end{pmatrix}, \tag{77}$$

$$\mathcal{M}_{\text{ind}} = m(y_p) + (K_{21})(K_{11})^{-1} \begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o) \end{pmatrix}, \tag{78}$$

where z_o consists of an observed subset of the components of the latent process z and z_p consists of another subset which is to be predicted. K_{ϵ_z} is the observation noise covariance matrix. Since z is a concatenation of initial value w_0 and wind velocity field u, K_{ϵ_z} is block-diagonal with Gaussian observation noise covariance matrices in its diagonal blocks. Similar in §5.1.2, we use the physics-based covariance model $\mathrm{Cov}(y_o, y_o) = L_o \mathrm{Cov}(z, z) L_o^T$ and $\mathrm{Cov}(y_p, y_o) = L_p \mathrm{Cov}(z, z) L_o^T$, where L_o, L_p correspond to the Jacobians of mapping F_o, F_p, respectively. The computation involving the three models is carried out using the low rank approximations. Finally we include kriging using the exact joint process covariance model (23)-(25) but without low rank approximations as contrast. We denote this exact joint process kriging model as $\mathcal{M}_e$.

For all the covariance models $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_{\text{ind}}$, $\mathcal{M}_e$ used here, each component of the latent process z is modeled by Gaussian process with square exponential covariance function, i.e. we assume

$$w_0 \sim \mathcal{N}(m(w_0), \alpha_w \exp(-\frac{r^2}{2l_w^2})), \tag{79}$$

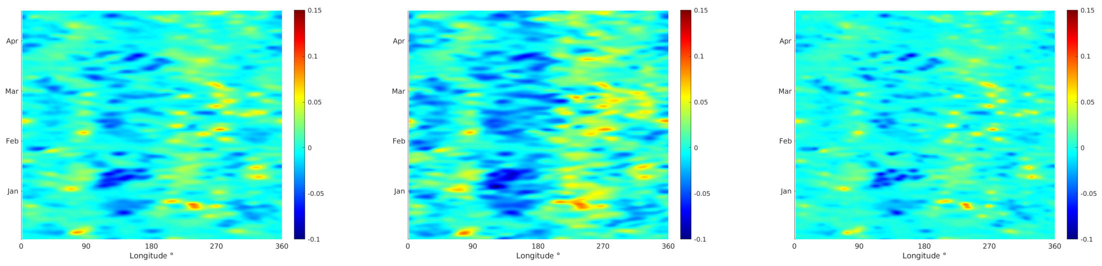
$$u \sim \mathcal{N}(m(u), \alpha_u \exp(-\frac{r^2}{2l_u^2})), \tag{80}$$

with magnitude parameters α_w, α_u and length scale parameters l_w, l_u. Let $\theta = (\alpha_w, \alpha_u, l_w, l_u)$ to be the unknown parameters that we will estimate and assume that the other parameters including model parameters in (75) and observation error covariance are known and specified at the beginning of §5.2 based on [36]. All the four parameters will be estimated from the observation data via score equations.

The processed relative humidity observation data are divided into different time snapshots with equal lengths. Each snapshot has 144×140 grid points (20160 data points) which represent 144 longitudes each day and an 140-day time window. First we randomly pick one snapshot field as calibration sample for parameter fitting. For models

TABLE 2: Predictive marginal log-likelihood values and RMS error (excluding observed locations). The observation density is 2.48% for q and 2.48% for u. The observations are randomly chosen in the field.

	Validation Sample		Calibration Sample	
Models	u_p	z_p	u_p	z_p
$\mathcal{M}_1$	0.0250	-	0.0207	-
$\mathcal{M}_2$	0.0238	0.0525	0.0208	0.0599
$\mathcal{M}_{\text{ind}}$	0.0266	0.0604	0.0226	0.0617
$\mathcal{M}_e$	0.0238	0.0523	0.0208	0.0598



(a) Latent process kriging $\mathcal{M}_1$ (b) Joint process kriging $\mathcal{M}_2$ (c) Joint independent kriging $\mathcal{M}_{\text{ind}}$

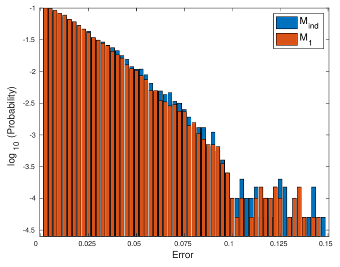
FIG. 6: Predictions corresponding to (a) Latent process kriging $\mathcal{M}_1$, (b) Joint process kriging $\mathcal{M}_2$ and (c) Joint independent kriging $\mathcal{M}_{\text{ind}}$. The time period is from Dec 1 1982 to Apr 18 1983. Both datasets are nondimensional.

$\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_{\text{ind}}$, we use scalable latent and joint process MLE described in §4. For model $\mathcal{M}_e$, we directly evaluate the score equations without approximations. The fitted parameters are used in the following kriging processes. The validation samples are chosen to be snapshots of the same size but at least 120 days away from the calibration sample, which can help to remove the effect of the trend of most intraseasonal oscillations. Numerical results are summarized in Table 2. We observe that (a) the approximation we made to the likelihood does not result in a significant kriging error, by comparing the validation sample performance of $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_e$ models whose performance is indistinguishable (despite the fact that model $\mathcal{M}_e$ is more expensive and not scalably computable) and (b) ignoring the covariance structure reduces the performance of the $\mathcal{M}_{ind}$ by about 20%. Therefore our approximation ensures scalability at a negligible cost to the accuracy; and the accurate modeling of the cross-covariance pays off in accuracy improvements for real data.

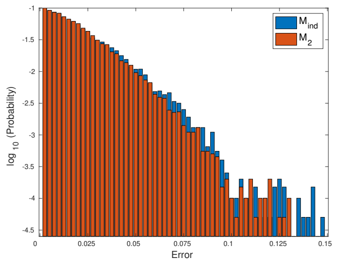
To provide a visual representation of the results, we illustrate in Figure 6 the relative humidity predictions on the calibration sample using model $\mathcal{M}_2$ and $\mathcal{M}_{\text{ind}}$. The predicted time window is Dec 1982 to Apr 1983 which is the same as observations in Figure 5. In Figure 7 we show histograms of the predictive errors of the two models respectively. Note that the vertical axis represents the probability density of a certain error range in logarithmic scale. For bars with zero probability, we truncate the logarithm and set it as the base value of the bar plot in order to prevent it from approaching negative infinity. As a result, the zero probability bars are also of height zero. We can observe the prediction error is reduced and has thinner tails when using the physics-based covariance models $\mathcal{M}_1$, $\mathcal{M}_2$, and particularly so for the latter.

We conclude that in this case the physics-based model produced a moderate improvement in prediction with this real data set. We have tried other configurations and while the physics-based cross-correlation model was consistently better than the independent model, the improvement seemed not particularly significant in several cases. There are several areas of improvement here, such as considering less smooth covariance kernels (at the price of increasing the rank of the approximation, or even changing the approximation strategy) and data sets with larger large scale variability or more pronounced nonlinear effects.

We note, however, that the computational performance was approximately linear with the number of data points, and the statistical performance was comparable to the one of exactly computed models which had far more intense



(a) Error histograms $\mathcal{M}_1$ vs. $\mathcal{M}_{\mathrm{ind}}$



(b) Error histograms $\mathcal{M}_2$ vs. $\mathcal{M}_{\mathrm{ind}}$

FIG. 7: Error histograms corresponding to independent process kriging $\mathcal{M}_{\mathrm{ind}}$ and joint process kriging $\mathcal{M}_2$. Note that the vertical axis represents the probability densities in logarithmic scale. The logarithm of zero density has been truncated to meet the base value of each bar plot.

computational requirements. We conclude that our approach of a physics-based implicitly defined kernel using our scalable approximation method brings a notable methodological improvement for real data sets, in that it is scalable as opposed to the classical approach and that it sometimes leads to noticeable improvements in the statistical performance.

6. DISCUSSION

Gaussian processes(GPs) are powerful tools in statistical modeling. In that space, the covariance structure takes a critical role in forecast accuracy and efficiency and there are few pointers on how to design good covariance kernels for complex processes. On the other hand, a great number of phenomena in natural sciences such as physics, biology, earth science, chemistry, have been well studied and mathematical or physical models have been established to interpret the underlying relations between the variables. Including such information from the physical models when constructing the covariance structure can be meaningful for forecasts when performing inferences on processes with partially known physical relations. However, one drawback of GPs is that their classical computational approach, based on the Cholesky factorization of an often-times dense kernel, scales cubically with number of observations. For large datasets, a reduction of computational cost is necessary before such methods become practical.

In this work, we utilize physics-based, implicitly-defined, covariance models and present a low-rank approximation of the covariance matrix that allows the GP regression and the MLE to be conducted in quasilinear time scale. The implicitly-defined nature of the kernel gives significant flexibility to the user who needs only access to a coarse physical model represented by a forward solver. Moreover, we also proposed a method to approximate the expected Fisher information matrix for quantifying the uncertainties of the estimates. Furthermore, we propose approaches to include higher-order terms in auto-covariance matrices that are essential for describing nonlinear processes while maintaining quasilinear scaling. In summary, we present a coherent framework for efficiently interpolating the random field and produced uncertainty estimates of this task by means of partial observations and exploiting approximate physical relationships. We presented several numerical experiments that demonstrate the accuracy of the approximated MLE by showing the approximated parameter estimates and the uncertainties are relatively close to the exact values when the latter are known. Then we used the estimated parameters in the physics-based covariance models and demonstrated that a covariance model that is complete and has correct physically consistent structure yields significant improvements in forecasts accuracy and efficiency. We then applied the framework to real climate data to further illustrate the effectiveness of our algorithms.

Our approach is implicit in that it only requires a black-box forward solver of the approximate physical relationships, making the extension of the algorithm to other models immediate. This feature increases the flexibility of the approach, benefiting from plentiful well-studied numerical discretization schemes and well-established simulation toolkits for solving physical models. Also our approach consists of independent calls to the solver which can fully take advantage of the parallel computing capability of modern hardware.

Our approach has several shortcomings. Our approximations of the gradient and Hessian can be accurate under certain regularity conditions of the physical model. For nondifferentiable or even discontinuous models, the approximations can be inefficient, making the approach case-dependent. Perhaps more problematic is the fact that we apply it to noise kernels which are very smooth. While this is a common choice, in many cases it may not be appropriate and we need to change the approximation method. We chose the Chebyshev interpolation method because it is simple to implement and the compressed covariance matrix is differentiable, but this can be improved by using the Nyström method and its variants [51,52], hierarchical matrix approximations [14,53], adaptive low-rank approximations [54,55]. In any case, simple global low-rank approximations as the ones we used here work well for covariance functions that are sufficiently smooth and dominated by long range relationships between data points. When the covariance function changes more rapidly, global low-rank methods cannot capture the variability and other methods, such as hierarchical matrix approximations [14,53] may be needed.

ACKNOWLEDGMENT

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research (ASCR) under Contract DE-AC02-06CH11347 and by NSF through award CNS-1545046.

APPENDIX A. JOINT PROCESS KRIGING WITH HIGHER-ORDER TERMS

Algorithm 6: Joint Process Kriging with Higher-order terms $\mathcal{M}_2^+$

Use the low-rank approximation with $N = O(\log n)$. Specifically,
$\mathrm{Cov}(z_o, z_o) \approx C_{z_o}^T K C_{z_o}, \mathrm{Cov}(z_o, z) \approx C_{z_o}^T K C_z, \mathrm{Cov}(z_p, z) \approx C_{z_p}^T K C_z, \mathrm{Cov}(z_p, z_o) \approx C_{z_p}^T K C_{z_o}.$

Step 1. Compute $A_1 = L_o C_z^T \in \mathbb{R}^{d\times N}$, $A_2 = L_p C_z^T \in \mathbb{R}^{(m-d)\times N}$ by $O(N)$ forward solves.

Step 2. Compute the Cholesky factorization $K = P^T P$, which takes $O(N^3)$ time.

Step 3. Draw $2N_l$ independent Rademacher vectors $\{u_i\}$,$\{v_i\} \in \mathbb{R}^N$, $i = 1, 2, \ldots, N_l$. Compute

$$\phi_i = C_z^T P^T u_i, \psi_i = C_z^T P^T v_i. \tag{A.1}$$

This takes $O(2N_l(N^2 + nN))$ time.

Step 4. Compute vector-Hessian-vector product by approximation (15). This approximation takes $O(N_l)$ forward solves. Denote $w_i = \psi_i^T H_o \phi_i$, $w_i' = \psi_i^T H_p \phi_i$. Then set the matrices
$A_3 \leftarrow \frac{1}{\sqrt{2N_l}}(w_1, w_2, \cdots, w_{N_l}) \in \mathbb{R}^{d\times N_l}$, $A_4 \leftarrow \frac{1}{\sqrt{2N_l}}(w_1', w_2', \cdots, w_{N_l}') \in \mathbb{R}^{(m-d)\times N_l}$. Using (37) we will approximate higher-order terms by

$$\frac{1}{2}\mathrm{tr}(H_o \mathrm{Cov}(\mathrm{z},\mathrm{z}) H_o^T \mathrm{Cov}(\mathrm{z},\mathrm{z})) \approx A_3 A_3^T, \tag{A.2}$$

$$\frac{1}{2}\mathrm{tr}(H_p \mathrm{Cov}(\mathrm{z},\mathrm{z}) H_o^T \mathrm{Cov}(\mathrm{z},\mathrm{z})) \approx A_4 A_3^T. \tag{A.3}$$

Step 5. Carry out the kriging computation (30) with components (31) and (32) in two steps. First, solve the modified inverse problem

$$\begin{aligned}\alpha &= \left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_3A_3^T & 0 \\ 0 & 0\end{pmatrix} + \begin{pmatrix} A_1 & 0 \\ 0 & C_{z_o}^T\end{pmatrix}\begin{pmatrix} K & K \\ K & K\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)^{-1}\begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o)\end{pmatrix}, \\ &= \left(\begin{pmatrix} K_{\epsilon_y} & 0 \\ 0 & K_{\epsilon_z}\end{pmatrix} + \begin{pmatrix} A_3 & A_1P^T \\ 0 & C_{z_o}^T P^T\end{pmatrix}\begin{pmatrix} A_3^T & 0 \\ PA_1^T & PC_{z_o}\end{pmatrix}\right)^{-1}\begin{pmatrix} y_o - m(y_o) \\ z_o - m(z_o)\end{pmatrix}\end{aligned} \tag{A.4}$$

by applying the SMW formula. The time cost is dominated by computing the matrix products
$\begin{pmatrix} A_3^T & 0 \\ PA_1^T & PC_{z_o}\end{pmatrix}\begin{pmatrix} K_{\epsilon_y}^{-1} & 0 \\ 0 & K_{\epsilon_z}^{-1}\end{pmatrix}\begin{pmatrix} A_3 & A_1P^T \\ 0 & C_{z_o}^T P^T\end{pmatrix}$ which take $O(N + N_l)$ linear system solves with the observation noise matrix and $O((n+m)(N+N_l)^2)$ assembly time.

Step 6. **Return** the final solution

$$\begin{pmatrix} m(y_p) \\ m(z_p)\end{pmatrix} + \left(\begin{pmatrix} A_4A_3^T & 0 \\ 0 & 0\end{pmatrix} + \begin{pmatrix} A_2 & 0 \\ 0 & C_{z_p}^T\end{pmatrix}\begin{pmatrix} K & K \\ K & K\end{pmatrix}\begin{pmatrix} A_1^T & 0 \\ 0 & C_{z_o}\end{pmatrix}\right)\alpha, \tag{A.5}$$

by matrix-vector product. It takes $O(2(n+m)N + 2mN_l + 4N^2)$ time.

In summary, based on our assumption of observation noise covariance matrices, the entire approach takes $O(N + N_l)$ forward solves, $O(N + N_l)$ observation noise covariance matrix linear system solves and the dominant computational cost takes $O((m+n)(N+N_l)^2)$ time, though the workflow is highly parallelizable. The cost is quasilinear with n if $N, N_l = log(n)$.

APPENDIX B. TENSOR ALGEBRA CONVENTIONS

Computing the higher-order terms in the physics-based covariance model requires tensor operations. We will use the tensor algebra with the following convention throughout the paper. Assume the vector-valued function F(z) defines a mapping $\mathbb{R}^n \to \mathbb{R}^m$, the Jacobian is an $m \times n$ matrix where each entry $L_{ij} = \frac{\partial F_i(z)}{\partial z_j}$. The Hessian of F is a rank-three tensor H. We arrange the order of indices of the tensor by $H_{ijk} = \frac{\partial^2 F_j(z)}{\partial y_i \partial y_k}$. Therefore, H is a $n \times m \times n$ tensor. The Hessian tensor can be viewed as an array of Hessian matrices of each components of the function:

$$H = \begin{pmatrix} H_1 \\ \vdots \\ H_m \end{pmatrix}, \tag{B.1}$$

where each $H_i \in \mathbb{R}^{n \times n}$ is the symmetric Hessian matrix of the i-th component of F. The transpose of the tensor is defined by permuting the first and last indices as well as the entire array:

$$H^T = (H_1^T, H_2^T, \cdots, H_m^T). \tag{B.2}$$

Following this convention, the tensor-tensor, tensor-vector and tensor-matrix product can be defined in an "element-wise sense": for any vector u and matrix U of proper size, we define

$$Hu = \begin{pmatrix} H_1 u \\ \vdots \\ H_m u \end{pmatrix}, \; HU = \begin{pmatrix} H_1 U \\ \vdots \\ H_m U \end{pmatrix}. \tag{B.3}$$

Additionally, we define the following tensor-tensor product which is useful in the quadratic form computation. Assume $\hat{H}$ is an $n \times \hat{m} \times n$ tensor,

$$HU\hat{H}^T U = \begin{pmatrix} H_1 U \hat{H}_1^T U & \cdots & H_1 U \hat{H}_{\hat{m}}^T U \\ \vdots & \ddots & \vdots \\ H_m U \hat{H}_1^T U & \cdots & H_m U \hat{H}_{\hat{m}}^T U \end{pmatrix}. \tag{B.4}$$

The left-multiplications are defined in the same element-wise sense, with left-multiplications in each entry instead. As a result, bilinear forms of such tensors on vectors $u, v, \in \mathbb{R}^m$ become a contraction along indices 1,3, and thus, the vector

$$v^T H u = \begin{pmatrix} v^T H_1 u \\ \vdots \\ v^T H_m u \end{pmatrix} \in \mathbb{R}^m. \tag{B.5}$$

The trace operation of the rank-three tensor is a tensor contraction to a vector taking over the first and last indices defined by

$$\mathrm{tr}(H) = [(\sum_{i,k} H_{ijk} \delta_{ik})_j] = \begin{pmatrix} \mathrm{tr}(H_1) \\ \vdots \\ \mathrm{tr}(H_m) \end{pmatrix}. \tag{B.6}$$

Particularly, the trace of rank-four tensor of form (B.4) can be defined

$$\mathrm{tr}(HU\hat{H}^T U) = \begin{pmatrix} \mathrm{tr}(H_1 U \hat{H}_1^T U) & \cdots & \mathrm{tr}(H_1 U \hat{H}_{\hat{m}}^T U) \\ \vdots & \ddots & \vdots \\ \mathrm{tr}(H_m U \hat{H}_1^T U) & \cdots & \mathrm{tr}(H_m U \hat{H}_{\hat{m}}^T U) \end{pmatrix} \in \mathbb{R}^{m \times \hat{m}}. \tag{B.7}$$

APPENDIX C. DERIVATION OF HIGHER-ORDER TERMS

Here the expectation of the quadratic form of centered multivariate Gaussian random variable is extensively used. Since $\delta z \in \mathbb{R}^n$ is a mean 0 Gaussian random variable, we have the following important properties as follows [56]: for any $n \times n$ symmetric matrix A, B,

$$\mathrm{E}(\delta z^T A \delta z) = \mathrm{tr}(A\mathrm{Cov}(z,z)) \tag{C.1}$$

$$\mathrm{E}[(\delta z^T A \delta z)(\delta z^T B \delta z)] = 2\mathrm{tr}(A\mathrm{Cov}(z,z)B\mathrm{Cov}(z,z)) + \mathrm{tr}(A\mathrm{Cov}(z,z))\mathrm{tr}(B\mathrm{Cov}(z,z)). \tag{C.2}$$

Proof. Since δz is a mean zero multivariate random variable,

$$\begin{aligned}\mathrm{E}(\delta z^T A \delta z) &= \mathrm{E}(\mathrm{tr}(\delta z^T A \delta z)) \\ &= \mathrm{E}(\mathrm{tr}(A \delta z \delta z^T)) \\ &= \mathrm{tr}(A\mathrm{E}(\delta z \delta z^T)) \\ &= \mathrm{tr}(A\mathrm{Cov}(z,z)).\end{aligned} \tag{C.3}$$

For equation (C.2), we start from simpler case. Assume $x \in \mathbb{R}^n$ is a standard normal random vector $x_i \sim \mathcal{N}(0,1)$ i.i.d. We first consider for any symmetric $n \times n$ matrices $\tilde{A}, \tilde{B}$,

$$\begin{aligned}\mathrm{E}(x^T \tilde{A} x x^T \tilde{B} x) &= \mathrm{E}\left(\sum_{i,j,k,l=1}^{n} \tilde{A}_{ij}\tilde{B}_{kl}x_i x_j x_k x_l\right) \\ &= \mathrm{E}\left(\sum_{i\neq j} \tilde{A}_{ii}\tilde{B}_{jj}x_i^2 x_j^2\right) + \mathrm{E}\left(\sum_{i\neq j} \tilde{A}_{ij}\tilde{B}_{ij}x_i^2 x_j^2\right) + \mathrm{E}\left(\sum_{i\neq j} \tilde{A}_{ij}\tilde{B}_{ji}x_i^2 x_j^2\right) + \mathrm{E}\left(\sum_{i=1}^{n} \tilde{A}_{ii}\tilde{B}_{ii}x_i^4\right) \\ &= \mathrm{E}\left(\sum_{i\neq j} \tilde{A}_{ii}\tilde{B}_{jj}\right) + 2\mathrm{E}\left(\sum_{i\neq j} \tilde{A}_{ij}\tilde{B}_{ji}\right) + 3\mathrm{E}\left(\sum_{i=1}^{n} \tilde{A}_{ii}\tilde{B}_{ii}\right)\end{aligned} \tag{C.4}$$

Note that the last step comes from the fact that $\tilde{B}$ is a symmetric matrix and the Isserlis' theorem. On the other hand, noticing that

$$\mathrm{tr}(\tilde{A}\tilde{B}) = \sum_{i,j=1}^{n} \tilde{A}_{ij}\tilde{B}_{ji} = \sum_{i\neq j}^{n} \tilde{A}_{ij}\tilde{B}_{ji} + \sum_{i=1}^{n} \tilde{A}_{ii}\tilde{B}_{ii} \tag{C.5}$$

$$\mathrm{tr}(\tilde{A})\mathrm{tr}(\tilde{B}) = \sum_{i\neq j}^{n} \tilde{A}_{ii}\tilde{B}_{jj} + \sum_{i=1}^{n} \tilde{A}_{ii}\tilde{B}_{ii}. \tag{C.6}$$

Comparing (C.4) with (C.5) and (C.6), then we have

$$\mathrm{E}(x^T \tilde{A} x x^T \tilde{B} x) = 2\mathrm{tr}(\tilde{A}\tilde{B}) + \mathrm{tr}(\tilde{A})\mathrm{tr}(\tilde{B}). \tag{C.7}$$

Now since δz is mean zero multivariate normal vector, there exist matrix L such that $\mathrm{Cov}(z,z) = LL^T$ and $\delta z = Lx$. Using (C.7) we have,

$$\begin{aligned}\mathrm{E}[(\delta z^T A \delta z)(\delta z^T B \delta z)] &= \mathrm{E}[(x^T(L^T A L)x)(x^T(L^T B L)x)] \\ &= 2\mathrm{tr}(L^T A L L^T B L) + \mathrm{tr}(L^T A L)\mathrm{tr}(L^T B L) \\ &= 2\mathrm{tr}(A L L^T B L L^T) + \mathrm{tr}(A L L^T)\mathrm{tr}(B L L^T) \\ &= 2\mathrm{tr}(A\mathrm{Cov}(z,z)B\mathrm{Cov}(z,z)) + \mathrm{tr}(A\mathrm{Cov}(z,z))\mathrm{tr}(B\mathrm{Cov}(z,z)).\end{aligned} \tag{C.8}$$

Henceforth we have proved (C.2).

□

Following our convention of tensor algebra for Hessian tensor H, we get

$$\begin{aligned}\overline{\delta z^T H \delta z} &= \mathrm{E}(\delta z^T H \delta z)\\ &= \begin{pmatrix}\mathrm{E}(\delta z^T H_1 \delta z)\\ \vdots \\ \mathrm{E}(\delta z^T H_m \delta z)\end{pmatrix}\\ &= \begin{pmatrix}\mathrm{tr}(H_1 \mathrm{Cov}(z,z))\\ \vdots \\ \mathrm{tr}(H_m \mathrm{Cov}(z,z))\end{pmatrix}\\ &= \mathrm{tr}(H\mathrm{Cov}(z,z)).\end{aligned} \tag{C.9}$$

Then for Hessian tensors $H \in \mathbb{R}^{n\times m\times n}, \hat{H} \in \mathbb{R}^{n\times \hat{m}\times n}$,

$$\overline{\delta z^T H \delta z}\;\overline{\delta z^T \hat{H}^T \delta z} = \mathrm{tr}(H\mathrm{Cov}(z,z))\mathrm{tr}(\hat{H}\mathrm{Cov}(z,z))^T. \tag{C.10}$$

One the other hand,

$$\begin{aligned}\overline{\delta z^T H \delta z \delta z^T \hat{H}^T \delta z} &= \begin{pmatrix}\mathrm{E}(\delta z^T H_1 \delta z \delta z^T \hat{H}_1^T \delta z) & \cdots & \mathrm{E}(\delta z^T H_1 \delta z \delta z^T \hat{H}_{\hat{m}}^T \delta z)\\ \vdots & \ddots & \vdots\\ \mathrm{E}(\delta z^T H_m \delta z \delta z^T \hat{H}_1^T \delta z) & \cdots & \mathrm{E}(\delta z^T H_m \delta z \delta z^T \hat{H}_{\hat{m}}^T \delta z)\end{pmatrix}\\ &\overset{(C.2)}{=} 2\mathrm{tr}(H\mathrm{Cov}(z,z)\hat{H}^T\mathrm{Cov}(z,z)) + \mathrm{tr}(H\mathrm{Cov}(z,z))\mathrm{tr}(\hat{H}\mathrm{Cov}(z,z))^T.\end{aligned} \tag{C.11}$$

REFERENCES

1. Rasmussen, C.E., Gaussian processes in machine learning, In *Summer School on Machine Learning*, pp. 63–71. Springer, 2003.
2. MacKay, D.J., Introduction to gaussian processes, *NATO ASI Series F Computer and Systems Sciences*, 168:133–166, 1998.
3. Williams, C.K. Prediction with gaussian processes: From linear regression to linear prediction and beyond. In *Learning in graphical models*, pp. 599–621. Springer, 1998.
4. Stein, M.L., *Interpolation of spatial data: some theory for kriging*, Springer Science & Business Media, 2012.
5. Kocijan, J., Murray-Smith, R., Rasmussen, C.E., and Girard, A., Gaussian process model based predictive control, In *Proceedings of the 2004 American control conference*, Vol. 3, pp. 2214–2219. IEEE, 2004.
6. Chiles, J.P. and Delfiner, P., *Geostatistics: modeling spatial uncertainty*, Vol. 497, John Wiley & Sons, 2009.
7. Boyle, P. and Frean, M., Dependent gaussian processes, In *Advances in neural information processing systems*, pp. 217–224, 2005.
8. Quiñonero-Candela, J. and Rasmussen, C.E., A unifying view of sparse approximate gaussian process regression, *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.
9. Tresp, V., A bayesian committee machine, *Neural computation*, 12(11):2719–2741, 2000.
10. Nguyen-Tuong, D., Peters, J.R., and Seeger, M., Local gaussian process regression for real time online model learning, In *Advances in Neural Information Processing Systems*, pp. 1193–1200, 2009.
11. Si, S., Hsieh, C.J., and Dhillon, I.S., Memory efficient kernel approximation, *The Journal of Machine Learning Research*, 18(1):682–713, 2017.
12. Xu, Z., Cambier, L., Rouet, F.H., L'Eplatennier, P., Huang, Y., Ashcraft, C., and Darve, E., Low-rank kernel matrix approximation using skeletonized interpolation with endo-or exo-vertices, *arXiv preprint arXiv:1807.04787*, 2018.
13. Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D.W., and O'Neil, M., Fast direct methods for gaussian processes, *IEEE transactions on pattern analysis and machine intelligence*, 38(2):252–265, 2015.

14. Börm, S. and Garcke, J., Approximating gaussian processes with $\mathcal{H}^2$-matrices, In *European Conference on Machine Learning*, pp. 42–53. Springer, 2007.

15. Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., and Rubin, D.B., *Bayesian data analysis*, Chapman and Hall/CRC, 2013.

16. Berliner, L.M. Hierarchical bayesian time series models. In *Maximum entropy and Bayesian methods*, pp. 15–22. Springer, 1996.

17. Wikle, C.K., Berliner, L.M., and Cressie, N., Hierarchical bayesian space-time models, *Environmental and Ecological Statistics*, 5(2):117–154, 1998.

18. Berliner, L.M., Royle, J.A., Wikle, C.K., and Milliff, R.F., Bayesian methods in the atmospheric sciences, *Bayesian statistics*, 6:83–100, 1999.

19. Royle, J., Berliner, L., Wikle, C., and Milliff, R. A hierarchical spatial model for constructing wind fields from scatterometer data in the labrador sea. In *Case Studies in Bayesian Statistics*, pp. 367–382. Springer, 1999.

20. Chilès, J.P. How to adapt kriging to non-classical problems: three case studies. In *Advanced geostatistics in the mining industry*, pp. 69–89. Springer, 1976.

21. Berliner, L.M., Physical-statistical modeling in geophysics, *Journal of Geophysical Research: Atmospheres*, 108(D24), 2003.

22. Berliner, L.M., Milliff, R.F., and Wikle, C.K., Bayesian hierarchical modeling of air-sea interaction, *Journal of Geophysical Research: Oceans*, 108(C4), 2003.

23. Wikle, C.K., Milliff, R.F., Nychka, D., and Berliner, L.M., Spatiotemporal hierarchical bayesian modeling tropical ocean surface winds, *Journal of the American Statistical Association*, 96(454):382–397, 2001.

24. Clark, J.S. and Gelfand, A.E., *Hierarchical modelling for the environmental sciences: statistical methods and applications*, Oxford University Press on Demand, 2006.

25. Constantinescu, E.M. and Anitescu, M., Physics-based covariance models for gaussian processes with multiple outputs, *International Journal for Uncertainty Quantification*, 3(1), 2013.

26. Yu, J. and Anitescu, M., Multidimensional sum-up rounding for integer programming in optimal experimental design, *Mathematical Programming*, pp. 1–40, 2019.

27. Xu, Z., Cambier, L., Rouet, F.H., L'Eplatennier, P., Huang, Y., Ashcraft, C., and Darve, E., Low-rank kernel matrix approximation using skeletonized interpolation with endo-or exo-vertices, *arXiv preprint arXiv:1807.04787*, 2018.

28. Alexanderian, A., Petra, N., Stadler, G., and Ghattas, O., A-optimal design of experiments for infinite-dimensional bayesian linear inverse problems with regularized \ell_0-sparsification, *SIAM Journal on Scientific Computing*, 36(5):A2122–A2148, 2014.

29. Mametjanov, A., Norris, B., Zeng, X., Drewniak, B., Utke, J., Anitescu, M., and Hovland, P. Applying automatic differentiation to the community land model. In *Recent Advances in Algorithmic Differentiation*, pp. 47–57. Springer, 2012.

30. Stein, M.L., Chen, J., Anitescu, M., , Stochastic approximation of score functions for gaussian processes, *The Annals of Applied Statistics*, 7(2):1162–1191, 2013.

31. Xu, W. and Anitescu, M., A limited-memory multiple shooting method for weakly constrained variational data assimilation, *SIAM Journal on Numerical Analysis*, 54(6):3300–3331, 2016.

32. Apte, A., Auroux, D., and Mythily, R., Variational data assimilation for discrete burgers equation, In *Electronic Journal of Differential Equations Conference*, pp. 15–30. Texas State Univ., 2010.

33. Kalnay, E., *Atmospheric modeling, data assimilation and predictability*, Cambridge university press, 2003.

34. Lewis, J.M., Lakshmivarahan, S., and Dhall, S., *Dynamic data assimilation: a least squares approach*, Vol. 13, Cambridge University Press, 2006.

35. Uboldi, F. and Kamachi, M., Time-space weak-constraint data assimilation for nonlinear models, *Tellus A*, 52(4):412–421, 2000.

36. Stechmann, S.N. and Hottovy, S., Unified spectrum of tropical rainfall and waves in a simple stochastic model, *Geophysical Research Letters*, 44(20):10–713, 2017.

37. Kloeden, P.E. and Platen, E., *Numerical solution of stochastic differential equations*, Vol. 23, Springer Science & Business Media, 2013.

38. Särkkä, S. and Solin, A., *Applied stochastic differential equations*, Vol. 10, Cambridge University Press, 2019.
39. Kalnay, E., Kanamitsu, M., Kistler, R., Collins, W., Deaven, D., Gandin, L., Iredell, M., Saha, S., White, G., Woollen, J., , The ncep/ncar 40-year reanalysis project, *Bulletin of the American meteorological Society*, 77(3):437–472, 1996.
40. Stechmann, S.N. and Ogrosky, H.R., The walker circulation, diabatic heating, and outgoing longwave radiation, *Geophysical Research Letters*, 41(24):9097–9105, 2014.
41. Stechmann, S.N. and Majda, A.J., Identifying the skeleton of the madden–julian oscillation in observational data, *Monthly Weather Review*, 143(1):395–416, 2015.
42. Ogrosky, H.R. and Stechmann, S.N., Identifying convectively coupled equatorial waves using theoretical wave eigenvectors, *Monthly Weather Review*, 144(6):2235–2264, 2016.
43. Waliser, D., Sperber, K., Hendon, H., Kim, D., Maloney, E., Wheeler, M., Weickmann, K., Zhang, C., Donner, L., Gottschalck, J., , Mjo simulation diagnostics, *Journal of Climate*, 22(11):3006–3030, 2009.
44. Chen, N. and Majda, A.J., Filtering the stochastic skeleton model for the madden–julian oscillation, *Monthly Weather Review*, 144(2):501–527, 2016.
45. Lagaris, I.E., Likas, A., and Fotiadis, D.I., Artificial neural networks for solving ordinary and partial differential equations, *IEEE transactions on neural networks*, 9(5):987–1000, 1998.
46. Khoo, Y., Lu, J., and Ying, L., Solving parametric pde problems with artificial neural networks, *arXiv preprint arXiv:1707.03351*, 2017.
47. Nabian, M.A. and Meidani, H., A deep neural network surrogate for high-dimensional random partial differential equations, *arXiv preprint arXiv:1806.02957*, 2018.
48. Anitescu, M., Chen, J., and Wang, L., A matrix-free approach for solving the parametric gaussian process maximum likelihood problem, *SIAM Journal on Scientific Computing*, 34(1):A240–A262, 2012.
49. Anitescu, M., Chen, J., and Stein, M.L., An inversion-free estimating equations approach for gaussian process models, *Journal of Computational and Graphical Statistics*, 26(1):98–107, 2017.
50. Geoga, C.J., Anitescu, M., and Stein, M.L., Scalable gaussian process computations using hierarchical matrices, *Journal of Computational and Graphical Statistics*, pp. 1–11, 2019.
51. Kumar, S., Mohri, M., and Talwalkar, A., Ensemble nystrom method, In *Advances in Neural Information Processing Systems*, pp. 1060–1068, 2009.
52. Wang, S. and Zhang, Z., Efficient algorithms and error analysis for the modified nystrom method, In *Artificial Intelligence and Statistics*, pp. 996–1004, 2014.
53. Ambikasaran, S., O'Neil, M., and Singh, K.R., Fast symmetric factorization of hierarchical matrices with applications, *arXiv preprint arXiv:1405.0223*, 2014.
54. Liberty, E., Woolfe, F., Martinsson, P.G., Rokhlin, V., and Tygert, M., Randomized algorithms for the low-rank approximation of matrices, *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.
55. Wang, S., Luo, L., and Zhang, Z., Spsd matrix approximation vis column selection: theories, algorithms, and extensions, *The Journal of Machine Learning Research*, 17(1):1697–1745, 2016.
56. Brookes, M., The matrix reference manual, *Imperial College London*, 3, 2005.