Multi-Fidelity High-Order Gaussian Processes for Physical Simulation

Zheng Wang, Wei Xing, Robert M. Kirby, Shandian Zhe
School of Computing, University of Utah
wzhut@cs.utah.edu,wxing@sci.utah.edu,kirby@cs.utah.edu,zhe@cs.utah.edu

Abstract

The key task of physical simulation is to solve partial differential equations (PDEs) on discretized domains, which is known to be costly. In particular, high-fidelity solutions are much more expensive than low-fidelity ones. To reduce the cost, we consider novel Gaussian process (GP) models that leverage simulation examples of different fidelities to predict high-dimensional PDE solution outputs. Existing GP methods are either not scalable to high-dimensional outputs or lack effective strategies to integrate multi-fidelity examples. To address these issues, we propose Multi-Fidelity High-Order Gaussian Process (MFHoGP) that can capture complex correlations both between the outputs and between the fidelities to enhance solution estimation, and scale to large numbers of outputs. Based on a novel nonlinear coregionalization model, MFHoGP propagates bases throughout fidelities to fuse information, and places a deep matrix GP prior over the basis weights to capture the (nonlinear) relationships across the fidelities. To improve inference efficiency and quality, we use bases decomposition to largely reduce the model parameters, and layer-wise matrix Gaussian posteriors to capture the posterior dependency and to simplify the computation. Our stochastic variational learning algorithm successfully handles millions of outputs without extra sparse approximations. We show the advantages of our method in several typical applications.

1 Introduction

Physical simulation (Keane and Nair, 2005) is critical for many science and engineering problems such as climate prediction and aircraft design. The core task of physical simulation is to solve partial differential equations (PDEs) for various physical models. Given the PDE parameters and initial/boundary conditions, traditional numerical solvers (Peiró and Sherwin, 2005) place a grid over the problem domain to discretize the PDEs and convert solving them into iteratively solving a linear system of equations. The solution field is represented by the solved function values at the grid points and hence are high-dimensional. Despite the success of traditional methods, they are known to be computationally costly (Santner et al., 2003). Even worse, any change of the PDE parameters or initial/boundary conditions will require re-computation from scratch (Oakley and O'Hagan, 2002). To reduce the cost, it is natural to consider using examples generated by the numerical solvers to train a machine learning model (Kennedy and O'Hagan, 2000), with which, we can directly predict the solution field (output) for new parameters and (parameterized) conditions (*i.e.,* input).

However, due to computational restrictions, the number of simulation examples is usually limited, and can be much smaller than the dimension of the solution output. Furthermore, collecting high-fidelity examples (with very accurate solution fields) is even more expensive, because we have to run the numerical solvers with very dense grids, which leads to an explosion in computation cost (Keane and Nair, 2005). In contrast, generating low-fidelity samples with coarser grids is cheaper, but low-fidelity samples can be quite inaccurate and biased. Hence, in practice we often can only obtain mixed examples where most are low-fidelity and only a few high-fidelity (Peherstorfer et al., 2018). Training with many low-fidelity examples can result in small variances but large biases, while training with very few high-fidelity samples can have small biases but much larger variances. To improve the prediction accuracy, it is crucial to effectively synergize and exploit the examples of all the fidelities.

To address this problem, we consider developing a novel Gaussian process (GP) model. While many excellent multi-output GPs can capture complex output correlations (Alvarez et al., 2012), they are often not scalable to high dimensional outputs and lack strategies to exploit multiple-fidelity samples to further improve training. Although Perdikaris et al. (2017) and Cutajar et al. (2019) fulfilled multi-fidelity

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

GP learning, they only estimate single output functions. While we can extend their work outright to a deep GP (Damianou and Lawrence, 2013) for multiple outputs, the outputs are fed into the next layer as the input of another GP and hence cannot be high-dimensional, say, hundreds of thousands or millions. In addition, the outputs in each layer are assumed to be independent given the inputs, so their strong dependencies might not be fully captured.

We propose MFHoGP, a multi-fidelity high-order GP model, which can capture the complex, strong correlations both between the fidelities and between the outputs to enhance function estimation, and efficiently scale up to large numbers of outputs. Our major contributions are listed as follows.

- We first propose a nonlinear coregionalization model for single-fidelity data. By introducing a matrix GP prior over the basis weights in the linear model of coregionalization (LMC) framework, our model is flexible enough to capture various nonlinear output correlations, while maintaining the scalability to high-dimensional outputs and a compact structure (*i.e.,* bases and weights) to enable efficient information propagation and fusion across different fidelities.
- Based on the nonlinear coregionalization, we propose a deep model to integrate multi-fidelity data. The model propagates bases throughout the fidelities, and uses a deep matrix GP prior to recursively sample the basis weights in each layer, so as to absorb the information from and capture the nonlinear relationship with the previous fidelities to further enhance function learning.
- We develop two simple yet effective tricks to improve inference efficiency and quality. First, we impose a decomposition structure upon the bases to greatly reduce the model parameters to save the computational cost and to avoid overfitting. Second, we propose a matrix Gaussian distribution as the variational posterior of the basis weights in each fidelity to capture their posterior dependency. The intrinsic Kronecker product structure further simplifies computation. We use the reparameterization trick to develop a stochastic variational learning algorithm that can handle millions of outputs without extra sparse approximations.

For evaluation, we first examined MFHoGP on small datasets to predict tens of thousands of outputs which correspond to solving classical Burgers', Poisson's and heat equations in small spatial/temporal domains. We trained MFHoGP with examples having one, two and three fidelities. In most cases, MFHoGP outperforms the state-of-the-art multi-output GP regression methods. The visualization of individual output prediction errors shows MFHoGP also better restores the outputs locally. Finally, we used MFHoGP to predict one million dimensional pressure fields of the lid-driven cavity flows, with only a few hundreds of training examples. By leveraging samples of two fidelities, our approach often achieves significant error reduction as compared with the single-fidelity competitors.

2 Background

The standard GP learns a single-output function $f : \mathbb{R}^s \to \mathbb{R}$ from the training data $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ where each $\mathbf{x}_n$ is an input vector. The function values $\mathbf{f} = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)]$ are assumed to follow a multivariate Gaussian distribution, $p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{K})$, where $\mathbf{m}$ are the mean function values of every input and usually set to $\mathbf{0}$, $[\mathbf{K}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is a kernel function of the input vectors. The observed outputs $\mathbf{y}$ are assumed to be sampled from a noisy model, *e.g.,* $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \tau\mathbf{I})$. Integrating out $\mathbf{f}$, we obtain the marginal likelihood $p(\mathbf{y}|\mathbf{X}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{nn} + \tau\mathbf{I})$. We can maximize the likelihood to estimate the kernel parameters and noise variance τ.

Many tasks require learning a function with multiple outputs. A classical multi-output regression framework is the Linear Model of Coregionalization (LMC) (Journel and Huijbregts, 1978), which assumes the outputs are a linear combination of a set of basis vectors weighted by independent random functions. We introduce K bases $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_K]^\top$ and model a d-dimensional vector function by

$$\mathbf{f}(\mathbf{x}) = \sum_{k=1}^{K} w_k(\mathbf{x})\mathbf{b}_k = \mathbf{B}^\top \mathbf{w}(\mathbf{x}) \tag{1}$$

where K is often chosen to be much smaller than d, and the random weight functions $\mathbf{w}(\mathbf{x}) = [w_1(\mathbf{x}), \ldots, w_K(\mathbf{x})]^\top$ are sampled from independent GPs. In spite of a linear structure, the outputs $\mathbf{f}(\mathbf{x})$ are still nonlinear to the input $\mathbf{x}$ due to the nonlinearity of the weight functions. LMC can easily scale up to a large number of outputs: once the bases $\mathbf{B}$ are identified, we only need to estimate a small number of GP models ($K \ll d$). For example, we can perform PCA on the training outputs to find the bases, and use the singular values as the outputs to train the weight functions. This is also referred to as PCA-GP (Higdon et al., 2008).

LMC is particularly useful for our physical simulation tasks because it is very efficient and scalable to high-dimensional solution outputs. Also, the compact structure — a small set of bases and weight functions — can be used to efficiently propagate and fuse information across multiple fidelities. Therefore, we will ground our work on LMC (other excellent models will be discussed in Sec. 5).

3 Model

A critical bottleneck of LMC is that it can only model linear correlations among the outputs (see the illustration mentioned below), which is oversimplified for physical simulation, where the high-dimensional solution outputs are governed by complex PDEs, implying strong nonlinear correlations. To fix this problem, one can place a GP prior over each element of the bases $\mathbf{B}$ (see (1)). This method is referred to as GP regression network (GPRN) (Wilson et al., 2012), and can greatly promote the flexibility to capture nonlinear output correlations. However, it will meanwhile

largely increase the computational cost— an extra Kd GP models need to be jointly estimated, which is very expensive for large d. Therefore, we propose a nonlinear generalization of LMC, which not only is flexible enough to capture nonlinear output correlations, but also maintains the efficiency and scalability to high-dimensional outputs. Based on the nonlinear generalization, we further develop a deep model to effectively integrate multi-fidelity data.

3.1 Nonlinear Coregionalization

The original LMC assumes independent random weight functions, which leads to oversimplified, linear output correlations. To see this, given an arbitrary input $\mathbf{x}$, we can derive the covariance of the outputs $\mathbf{f}(\mathbf{x})$ according to (1): $\text{cov}(\mathbf{f}) = \mathbf{B}^\top \text{cov}\big(\mathbf{w}(\mathbf{x})\big)\mathbf{B}$. Since the weight functions $\mathbf{w}(\mathbf{x})$ are sampled independently, $\text{cov}\big(\mathbf{w}(\mathbf{x})\big)$ must be diagonal, and therefore $\text{cov}(\mathbf{f})$ is essentially a $d \times d$ linear kernel matrix on $\mathbf{B}^\top$ (which is $d \times k$), implying linear correlations.

To grasp the nonlinear output correlations, we break the independent assumption of the weight functions. Instead, we consider the weights also as a nonlinear function of the bases, and model their correlations with a nonlinear kernel of the bases (*e.g.,* RBF and Matern). To this end, we jointly sample the K weight functions from a matrix-variate GP. Given N training inputs $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top$ and K bases $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_K]^\top$, the weight functions' projection $\mathbf{W}$ (which is an $N \times K$ matrix and each element $[\mathbf{W}]_{ij} = w_j(\mathbf{x}_i)$) then follows a matrix Gaussian distribution,

$$p(\mathbf{W}|\mathbf{X}, \mathbf{B}) = \mathcal{MN}(\mathbf{W}|\mathbf{0}, \mathbf{K}, \mathbf{K}_{BB}), \tag{2}$$

where the row-covariance $\mathbf{K}$ is the kernel matrix on the inputs $\mathbf{X}$, $[\mathbf{K}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, and the column-covariance $\mathbf{K}_{BB}$ the kernel matrix on the bases $\mathbf{B}$, $[\mathbf{K}_{BB}]_{mt} = k_b(\mathbf{b}_m, \mathbf{b}_t)$. Given the weights and bases, we sample the observed $N \times d$ output matrix $\mathbf{Y}$ from a Gaussian noise model, $p(\mathbf{Y}|\mathbf{W}, \mathbf{B}) = \mathcal{N}\big(\text{vec}(\mathbf{Y})|\text{vec}(\mathbf{WB}), \eta^{-1}\mathbf{I}\big)$, where $\text{vec}(\cdot)$ is the vectorization and η the inverse noise variance. This new model, referred to as nonlinear coregionalization, turns out to be a GP model.

Lemma 3.1. *The marginal distribution of the output $\mathbf{Y}$ is*

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{B}) = \mathcal{N}\big(\text{vec}(\mathbf{Y})|\mathbf{0}, (\mathbf{B}^\top\mathbf{K}_{BB}\mathbf{B}) \otimes \mathbf{K} + \eta^{-1}\mathbf{I}\big).$$

Given two arbitrary outputs $y_m(\mathbf{x}_i)$ and $y_t(\mathbf{x}_j)$, i.e., the m-th output for input $\mathbf{x}_i$ and t-th output for input $\mathbf{x}_j$, we have $\text{cov}\big(y_m(\mathbf{x}_i), y_t(\mathbf{x}_j)\big) = k(\mathbf{x}_i, \mathbf{x}_j)\tilde{b}_m^\top\mathbf{K}_{BB}\tilde{b}_t + \eta^{-1}\cdot\delta(\mathbf{x}_i = \mathbf{x}_j, m = t)$, where $\tilde{b}_m$ and $\tilde{b}_t$ are the m-th and t-th column of $\mathbf{B}$, respectively, and $\delta(\cdot)$ is the indicator function.

The proof is given in the supplementary material. Now, we can see that given any input $\mathbf{x}$, $\text{cov}\big(\mathbf{y}(\mathbf{x})\big) = k(\mathbf{x}, \mathbf{x})\mathbf{B}^\top\mathbf{K}_{BB}\mathbf{B} + \eta^{-1}\mathbf{I}$. As long as $\mathbf{K}_{BB}$ is constructed from a nonlinear kernel, the covariance matrix is nonlinear to the bases and so are the output correlations. The LMC can be viewed as an instance of our model with a particular choice of the bases' kernel.

Corollary 3.1.1. *When we set the bases' kernel $k_b(\mathbf{b}_m, \mathbf{b}_t) = \delta(\mathbf{b}_m = \mathbf{b}_t)$, the model is reduced to LMC with the same kernel for all the weight functions.*

Note that by placing a matrix GP prior over $\mathbf{W}$, we enable LMC to capture nonlinear output dependencies, without the need for any additional latent functions (like GPRN). By exploiting the inherent Kronecker product (see Lemma 3.1), we can further simplify the computation to avoid calculating the full covariance matrix (Stegle et al., 2011). The extra calculation only involves one small $K \times K$ kernel matrix on the bases, namely $\mathbf{K}_{BB}$ (in practice, K is usually chosen to be less than N (Higdon et al., 2008)). By contrast, GPRN places a GP prior over every element of $\mathbf{B}$ and hence needs to compute/estimate Kd prior/posterior covariance matrices of all the latent functions in $\mathbf{B}$, which will be very expensive for large d, *e.g.,* millions ($\mathcal{O}(N^3Kd)$ time complexity).

3.2 Multi-Fidelity Nonlinear Coregionalization

Next, to exploit training samples with multiple fidelities, we use the nonlinear coregionalization as the basic component and propose a deep model that propagates bases and places a deep matrix GP prior over the weight functions in all the fidelities. In each level, we use one component to sample the observed outputs in a particular fidelity. Each component inherits the bases from and samples the weight functions conditioned on the weights of the previous level. In this way, we capture the (nonlinear) relationships with and reuse the valuable bases from previous fidelities to enhance the predictions for the current fidelity.

Specifically, suppose we have training examples of F fidelities, $\{(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)})\}_{i=1}^F$ where $\mathbf{X}^{(i)}$ and $\mathbf{Y}^{(i)}$ are the $N_i \times s$ input and $N_i \times d$ output matrices at fidelity i. Note that although the solutions of different fidelities are calculated from distinct grids, we align them to the same dimension with a fixed grid via interpolation (note that it does not influence the fidelity) (Zienkiewicz et al., 1977). Fidelity i is lower than its successive fidelity $i+1$ and hence $N_1 > \dots > N_F$. Following the standard multi-fidelity simulation setting (Perdikaris et al., 2017; Peherstorfer et al., 2018), we assume the inputs of higher fidelity samples are a subset of the lower fidelity ones, *i.e.,* $\mathbf{X}^{(F)} \subset \dots \subset \mathbf{X}^{(1)}$. However, our method can be trivially adjusted for non-overlapping inputs (see the discussion in Sec. 3 of the supplementary material). Denote by $\mathbf{W}^{(i)}$ and $\mathbf{B}^{(i)}$ the bases and weights in each fidelity i. We sample the output matrix $\mathbf{Y}^{(i)}$ from $p(\mathbf{Y}^{(i)}|\mathbf{W}^{(i)}, \mathbf{B}^{(i)}, \{\eta_j\}_{j=1}^i) = \mathcal{N}(\text{vec}(\mathbf{Y}^{(i)})|\text{vec}(\mathbf{W}^{(i)} \cdot \mathbf{B}^{(i)}), \prod_{j=1}^i \eta_j^{-1} \cdot \mathbf{I})$, where each η_j is independently sampled from a Gamma prior, $p(\eta_j) = \text{Gamma}(\eta_j|\alpha, 1)$ where $\alpha > 1$. Note that we use a product of Gamma random variables as the inverse variance to gradually diminish the noise level with the increase of the fidelity. This is consistent with the fact that samples of higher fidelities should be more accurate and less noisy.

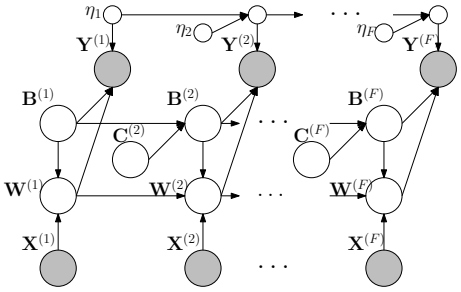


Figure 1: The graphical representation of MFHoGP.

In the first (lowest) fidelity ($i = 1$), we sample the bases $\mathbf{B}^{(1)}$ from a continuous prior, say, Gaussian, and the weights $\mathbf{W}^{(1)}$ from the matrix GP prior in (2). In each higher fidelity ($i > 1$), we inherit the bases from the previous level, and sample additional K bases $\mathbf{C}^{(i)}$ from the continuous prior again. We combine $\mathbf{B}^{(i-1)}$ and $\mathbf{C}^{(i)}$ to construct the bases for the current fidelity, $\mathbf{B}^{(i)} = [\mathbf{B}^{(i-1)}; \mathbf{C}^{(i)}]$. In this way, we take advantage of not only the valuable bases from the previous fidelities — an effective summary of lower fidelities' information, but also the ones specific to the current fidelity. Between fidelities can be complex yet strong relationships. To capture and exploit these relationships, we involve the weights of the previous fidelity in generating the weights of the current fidelity. Specifically, we append to the current inputs $\mathbf{X}^{(i)}$ the corresponding basis weights of the previous fidelity, $\widehat{\mathbf{X}}^{(i)} = [\mathbf{X}^{(i)}, \mathbf{W}^{(i-1)}(\mathbf{X}^{(i)}, \mathbf{B}^{(i-1)})]$. We then sample $\mathbf{W}^{(i)}$ from a matrix GP prior similar to (2),

$$p(\mathbf{W}^{(i)}|\mathbf{B}^{(i)}, \mathbf{X}^{(i)}, \mathbf{W}^{(i-1)}) = p(\mathbf{W}^{(i)}|\mathbf{B}^{(i)}, \widehat{\mathbf{X}}^{(i)}) = \mathcal{MN}(\mathbf{W}^{(i)}|\mathbf{0}, \mathbf{K}^{(i)}, \mathbf{K}^{(i)}_{BB}) \quad (3)$$

where $\mathbf{K}^{(i)}$ is the kernel matrix on the augmented inputs $\widehat{\mathbf{X}}^{(i)}$ and $\mathbf{K}^{(i)}_{BB}$ the kernel matrix on $\mathbf{B}^{(i)}$. The chain of the matrix GPs hence forms a deep matrix GP prior over all the weight functions $\{\mathbf{W}^{(i)}\}_{i=1}^F$ to capture the (nonlinear) relationships across the fidelities. Finally, the graphical representation of our model is given in Fig. 1.

4 Algorithm

For efficient model estimation, we develop a stochastic variational learning algorithm that jointly updates the bases $\mathbf{B}$, the kernel and noise parameters $\{\eta_i\}$, and the variational posterior of the weight functions $\{\mathbf{W}^{(i)}\}$.

4.1 Decomposition Structure for Bases

First, we introduce bases decomposition to further reduce the model parameters, the computation cost and also to avoid overfitting. In practice, the output dimension d can be very large, say, millions. Since each basis in $\{\mathbf{B}^{(i)}\}_{i=1}^F$ is a d dimensional vector, it will introduce too many parameters. The estimation of these parameters will be costly and the model can easily overfit the (small) data. To overcome these problems, we impose a decomposition structure on the bases to greatly reduce the parameters. Specifically, for each basis $\mathbf{b}^i_j$ in fidelity i (note that $\mathbf{B}^{(i)} = [\mathbf{b}^i_1, \ldots, \mathbf{b}^i_K]^\top$), we introduce R compositional vectors, $U^i_j = \{\mathbf{u}^i_{j1}, \ldots, \mathbf{u}^i_{jR}\}$, each with length $\sqrt[R]{d}$, and parameterize $\mathbf{b}^i_j = \mathbf{u}^i_{j1} \otimes \ldots \otimes \mathbf{u}^i_{jR}$ where $\otimes$ is the Kronecker product. The kernel function of two bases $\mathbf{b}^i_{j_1}$ and $\mathbf{b}^i_{j_2}$ is then defined on their compositional vectors, $k_b(\mathbf{b}^i_{j_1}, \mathbf{b}^i_{j_2}) = k_b(U^i_{j_1}, U^i_{j_2})$. Take $d = 10^6$ as an example. If we choose $R = 3$, we only need to use three 100 dimensional compositional vectors to calculate each basis, and the parameters are reduced by 99.97%. The proposed structure is essentially a rank-1 CP (Harshman, 1970) decomposition on the tensorized basis with R modes. We can also use higher ranks or other decomposition structures, but this simple structure has already shown the advantages of our model in the experiments (see Sec. 6).

We assign a standard Gaussian prior over each compositional vector, $p(\mathbf{u}^i_j) = \mathcal{N}(\mathbf{u}^i_j|\mathbf{0}, \mathbf{I})$. We then parameterize each row of $\mathbf{B}^{(i)}$ and $\mathbf{C}^{(i)}$ by the Kronecker product of their corresponding compositional vectors. Note that the bases $\mathbf{B}^{(i)}$ are still constructed as $[\mathbf{B}^{(i-1)}; \mathbf{C}^{(i)}]$ when $i > 1$. Denote the compositional vectors in each fidelity i by $\mathcal{U}^{(i)} = \{U^i_1, \ldots, U^i_K\}$. The joint probability now is

$$p(\{\mathcal{U}^{(i)}, \mathbf{W}^{(i)}, \mathbf{Y}^{(i)}, \eta_i\}_{i=1}^F | \{\mathbf{X}^{(i)}\}_{i=1}^F) = \prod_{i=1}^F \mathrm{Gam}(\eta_j|\alpha, 1) \cdot \prod_{i=1}^F \prod_{j=1}^K \prod_{r=1}^R \mathcal{N}(\mathbf{u}^i_{jr}|\mathbf{0}, \mathbf{I}) \prod_{i=1}^F \mathcal{MN}(\mathbf{W}^{(i)}|\mathbf{0}, \mathbf{K}^{(i)}, \mathbf{K}^{(i)}_{BB}) \cdot \prod_{i=1}^F \mathcal{N}\big(\mathrm{vec}(\mathbf{Y}^{(i)})|\mathrm{vec}(\mathbf{W}^{(i)}\mathbf{B}^{(i)}), \prod_{j=1}^i \eta_j^{-1}\mathbf{I}\big). \quad (4)$$

The model inference amounts to estimating the compositional vectors $\{\mathcal{U}^{(i)}\}$ for the bases, the posteriors of the weight functions in each fidelity and other parameters.

4.2 Layer-Wise Matrix Gaussian Posterior

Next, we introduce a variational posterior for the weight functions in all the fidelities $\mathcal{W} = \{\mathbf{W}^{(i)}\}_{i=1}^F$ and construct a variational model evidence lower bound, $\mathcal{L} = \mathbb{E}_{q(\mathcal{W})}\big[\log\big(p(\{\mathcal{U}^{(i)}, \mathbf{W}^{(i)}, \mathbf{Y}^{(i)}, \eta_i\}_{i=1}^F | \{\mathbf{X}^{(i)}\}_{i=1}^F)\big) - \log(q(\mathcal{W}))\big]$. While we can follow the standard mean-field framework to use fully independent posteriors, they will break the strong posterior dependency among the weights, and may result in inferior inference quality. Note that the matrix GP prior of each $\mathbf{W}^{(i)}$ (see (2) (3)) has incorporated (nonlinear) correlations between the weight functions. To capture the posterior dependency, we introduce a matrix Gaussian distribution as the variational posterior of each $\mathbf{W}^{(i)}$, consistent with the prior. The variational posterior of all the weights $\mathcal{W}$ is then given by

$$q(\mathcal{W}) = \prod_{i=1}^F q(\mathbf{W}^{(i)}) = \prod_{i=1}^F \mathcal{MN}(\mathbf{W}^{(i)}|\mathbf{M}^{(i)}, \mathbf{\Sigma}^{(i)}, \mathbf{\Omega}^{(i)}),$$

where $\mathbf{M}^{(i)}$, $\mathbf{\Sigma}^{(i)}$ and $\mathbf{\Omega}^{(i)}$ are the posterior mean, row and column covariances of each $\mathbf{W}^{(i)}$. Another advantage is the computational efficiency. Due to the intrinsic Kronecker product, we never need to compute the full covariance matrix of the density (Stegle et al., 2011). Instead, it can be calculated by the row and column covariance matrices and hence the cost is largely reduced, $q(\mathbf{W}^{(i)}) = \mathcal{MN}(\mathbf{W}^{(i)}|\mathbf{M}^{(i)}, \mathbf{\Sigma}^{(i)}, \mathbf{\Omega}^{(i)}) = \mathcal{N}\big(\mathrm{vec}(\mathbf{W}^{(i)})|\mathrm{vec}(\mathbf{M}^{(i)}), \mathbf{\Omega}^{(i)} \otimes \mathbf{\Sigma}^{(i)}\big) = \exp\big(-\frac{1}{2}\mathrm{tr}[{\mathbf{\Omega}^{(i)}}^{-1}(\mathbf{W}^{(i)} - \mathbf{M}^{(i)})^\top {\mathbf{\Sigma}^{(i)}}^{-1}(\mathbf{W}^{(i)} - \mathbf{M}^{(i)})]\big)/\big((2\pi)^{iNK/2}|\mathbf{\Omega}^{(i)}|^{N_i/2}|\mathbf{\Sigma}^{(i)}|^{iK/2}\big)$. Note that the same computation applies to the prior of $\{\mathbf{W}^{(i)}\}$. We derive the variational evidence lower bound (ELBO) finally,

$$\begin{aligned}\mathcal{L} = &\sum_{i=1}^{F}\sum_{j=1}^{K}\sum_{r=1}^{R} -\frac{1}{2}\|\mathbf{u}_{jr}^i\|^2 + \sum_{j=1}^{i}\frac{N_i}{2}\big(d\log\eta_i - \log|\mathbf{K}_{BB}^{(i)}|\big)\\ &-\frac{1}{2}\sum_{i=1}^{F} iK\log|\mathbf{\Sigma}^{(i)}| + (\prod_{j=1}^{i}\eta_j)\mathrm{tr}(\mathbf{\Sigma}^{(i)})\mathrm{tr}(\mathbf{\Omega}^{(i)}\mathbf{B}^{(i)}\mathbf{B}^{(i)^\top})\\ &+\frac{1}{2}\sum_{i=1}^{F} N_i\log|\mathbf{\Omega}^{(i)}| - (\prod_{j=1}^{i}\eta_j)\big(\|\mathbf{Y}_i - \mathbf{M}^{(i)}\mathbf{B}^{(i)}\|_{\mathcal{F}}^2\big) \quad (5)\\ &+\sum_{i=1}^{F}(\alpha-1)\log\eta_i - \eta_i - \frac{1}{2}\mathbb{E}_{q(\mathcal{W})}\big[iK\log(\mathbf{K}^{(i)})\big]\\ &-\frac{1}{2}\sum_{i=1}^{F}\mathbb{E}_{q(\mathcal{W})}\big[\mathrm{tr}\big({\mathbf{K}_{BB}^{(i)}}^{-1}{\mathbf{W}^{(i)}}^\top{\mathbf{K}^{(i)}}^{-1}\mathbf{W}^{(i)}\big)\big] + \mathrm{const.}\end{aligned}$$

where $\|\cdot\|_{\mathcal{F}}$ is the Frobenius norm.

4.3 Stochastic Optimization

We aim to maximize the variational ELBO in (5). However, the expectation terms involving each $\mathbf{K}^{(i)}$ are intractable, because they are kernel matrices on the augmented inputs $\widehat{\mathbf{X}}^{(i)} = [\mathbf{X}^{(i)}, \mathbf{W}^{(i-1)}(\mathbf{X}^{(i)}, \mathbf{B}^{(i-1)})]$, where the weights from the previous fidelity are (partly) coupled in the nonlinear kernels. To address this issue, we use the reparameterization trick (Kingma and Welling, 2013) to calculate an unbiased stochastic gradient for optimization. In each fidelity i, we sample a standard matrix Gaussian random variable, $\mathbf{Z}^{(i)} \sim \mathcal{MN}(\mathbf{Z}^{(i)}|\mathbf{0}, \mathbf{I}, \mathbf{I})$. Then we construct a parameterized sample, $\widetilde{\mathbf{W}}^{(i)} = \mathbf{M}^{(i)} + \mathbf{L}^{(i)}\mathbf{Z}^{(i)}{\mathbf{R}^{(i)}}^\top$, where $\mathbf{L}^{(i)}$ and $\mathbf{R}^{(i)}$ are the Cholesky decompositions of the row covariance $\mathbf{\Sigma}^{(i)}$ and column covariance $\mathbf{\Omega}^{(i)}$ in $q(\mathbf{W}^{(i)})$, respectively. According to the following theorem, $\widetilde{\mathbf{W}}^{(i)}$ is guaranteed to be a sample of $q(\mathbf{W}^{(i)})$.

Theorem 4.1. *(Gupta and Nagar, 1999) Given $n \times p$ matrix $\mathbf{Z}$, $m \times n$ matrix $\mathbf{G}$ and $p \times l$ matrix $\mathbf{H}$, If $\mathbf{Z} \sim \mathcal{MN}(\cdot|\mathbf{A}, \mathbf{\Sigma}, \mathbf{\Psi})$, $\mathrm{rank}(\mathbf{G}) \le n$ and $\mathrm{rank}(\mathbf{H}) \le p$, then $\mathbf{LZR} \sim \mathcal{MN}(\cdot|\mathbf{G\Sigma G}^\top, \mathbf{H}^\top\mathbf{\Psi H})$.*

Corollary 4.1.1. *The constructed sample $\widetilde{\mathbf{W}}^{(i)} \sim \mathcal{MN}(\cdot|\mathbf{M}^{(i)}, \mathbf{\Sigma}^{(i)}, \mathbf{\Omega}^{(i)})$, namely $q(\mathbf{W}^{(i)})$.*

Next, we sequentially append each $\widetilde{\mathbf{W}}^{(i-1)}(\mathbf{X}^{(i)}, \mathbf{B}^{(i-1)})$ to $\mathbf{X}^{(i)}$ to obtain the augmented inputs, based on which we compute the random kernel matrix $\widetilde{\mathbf{K}}^{(i)}$ ($i > 1$). We then replace each $\mathbb{E}_{q(\mathcal{W})}\big[\mathrm{tr}\big({\mathbf{K}_{BB}^{(i)}}^{-1}{\mathbf{W}^{(i)}}^\top{\mathbf{K}^{(i)}}^{-1}\mathbf{W}^{(i)}\big)\big]$ and $\mathbb{E}_{q(\mathcal{W})}\big[iK\log(\mathbf{K}^{(i)})\big]$ in (5) with their unbiased estimates $\mathrm{tr}\big({\mathbf{K}_{BB}^{(i)}}^{-1}{\widetilde{\mathbf{W}}^{(i)}}^\top{\widetilde{\mathbf{K}}^{(i)}}^{-1}\widetilde{\mathbf{W}}^{(i)}\big)$ and $iK\log(\widetilde{\mathbf{K}}^{(i)})$, respectively, so as to obtain an unbiased stochastic bound $\widetilde{\mathcal{L}}$. We compute $\nabla\widetilde{\mathcal{L}}$ as an unbiased stochastic gradient of $\mathcal{L}$ for optimization. We can use any stochastic optimization algorithm to jointly update the basis compositional vectors $\{\mathcal{U}^{(i)}\}$, the variational posterior $q(\mathcal{W})$ (determined by $\{\mathbf{M}^{(i)}, \mathbf{L}^{(i)}, \mathbf{R}^{(i)}\}$) and all the other parameters.

4.4 Prediction

Given a new input, the predictive distribution of the outputs is not analytical. Hence, we recursively sample the weights in each fidelity to generate posterior samples, with which we compute an empirical distribution. Due to the space limit, we leave the details in the supplementary material.

4.5 Algorithm Complexity

The time complexity of our inference algorithm is $\mathcal{O}(\sum_{i=1}^{F} iN_iKd + (iK)^2iRd^{\frac{1}{R}} + (iK)^3 + N_i^3)$. Since we can always choose R such that $Rd^{\frac{1}{R}} \le d$ (the simplest choice is $R = 1$), the time complexity is linear to Nd, where N is the total number of samples. The space complexity is $\mathcal{O}(FKd + \sum_{i=1}^{F} iN_iK + (iK)^2 + (N_i)^2)$, including the storage of the bases, the weights, and the row and column covariance matrices of the weights in each fidelity.

5 Related Work

Many multi-output GP regression approaches have been proposed. An excellent review is given in (Alvarez et al., 2012). A classical framework is the linear model of coregionalization (LMC) (Matheron, 1982; Goulard and Voltz, 1992), which introduces a set of basis vectors, and use their linear combination weighted by independent random functions to predict the output vector. A popular instance is PCA-GP (Higdon et al., 2008) that finds a set of bases from Singular Value Decomposition (SVD) on the training outputs. The variants of PCA-GP include KPCA-GP (Xing et al., 2016), IsoMap-GP (Xing et al., 2015), *etc.* Despite its efficiency and scalability, the standard LMC only models linear output correlations. GP regression networks (GPRNs) (Wilson et al., 2012) overcome this problem by placing independent GP priors over the basis elements. While being much more expressive, GPRNs bring in much more computation cost — the number of GPs need to be estimated is quite a few times (*e.g.,* tens) of the output dimension, and hence it will be very expensive for high dimensions. Important multi-output GP models also include convolved GPs (Higdon, 2002; Boyle and Frean, 2005; Alvarez et al., 2019) and multi-task GPs (Bonilla et al., 2007, 2008; Rakitsch et al., 2013). Both types of models are very elegant and flexible, however, they might be computationally too costly ($\mathcal{O}((Nd)^3)$ or $\mathcal{O}(N^3 + d^3)$ time complexity) for massive outputs. To mitigate this issue, several sparse approxima-

tions have been developed (Alvarez and Lawrence, 2009; Álvarez et al., 2010). Recently, Zhe et al. (2019) tensorized the high dimensional output, and introduced latent coordinate features in the tensor space to model complex output correlations. Overall, all these methods are developed for single-fidelity data.

To enable GP training on multi-fidelity data, Perdikaris et al. (2017) sequentially learned a chain of GPs, where each GP estimates the output of one fidelity as a function of the current input and the output of the previous fidelity. Cutajar et al. (2019) jointly learned these GP models to propagate the uncertainty throughout different fidelities. These excellent works focus on single output functions. While we can extend them to a standard deep GP (Damianou and Lawrence, 2013; Hebbal et al., 2019) that samples multiple functions in each layer, all the outputs in one layer are poured as the input to the GP in the next layer, and hence cannot be many, say, millions. Moreover, standard deep GPs consider the outputs in each layer as independent given the inputs, and might not fully capture the strong output dependencies, which is crucial for learning from a small set of training examples (in physical simulation). To address these problems, we inherit the compact structure of LMC, *i.e.,* a small number of bases and weight functions to handle massive outputs. We first generalize LMC to flexibly capture nonlinear output correlations. We then propagate the (decomposed) bases and place a deep matrix GP prior over the weight functions to fuse information throughout the fidelities (rather than use the entire outputs), and hence it is much more efficient. Recently, Hamelijnck et al. (2019) proposed a multi-task multi-resolution GP model based on GPRN, deep GP and mixture of experts (Rasmussen and Ghahramani, 2002). This work aims to integrate sensor data with different resolutions. Distinct from our model, it needs to integrate over the sampling periods of the sensors to sample the observations, and emphasizes one particular task (output). It does not scale to many outputs.

Recently, a few excellent works were proposed to learn (deep) neural networks to solve PDEs (Raissi, 2018; Raissi et al., 2019). These works differ from ours in that (1) the input is the spatial/temporal location and the output is a scalar to predict the solution function value at that location, and (2) their training and test focus on solving one particular PDE, rather than mapping parameters of different PDEs to their corresponding solution fields at a specific grid.

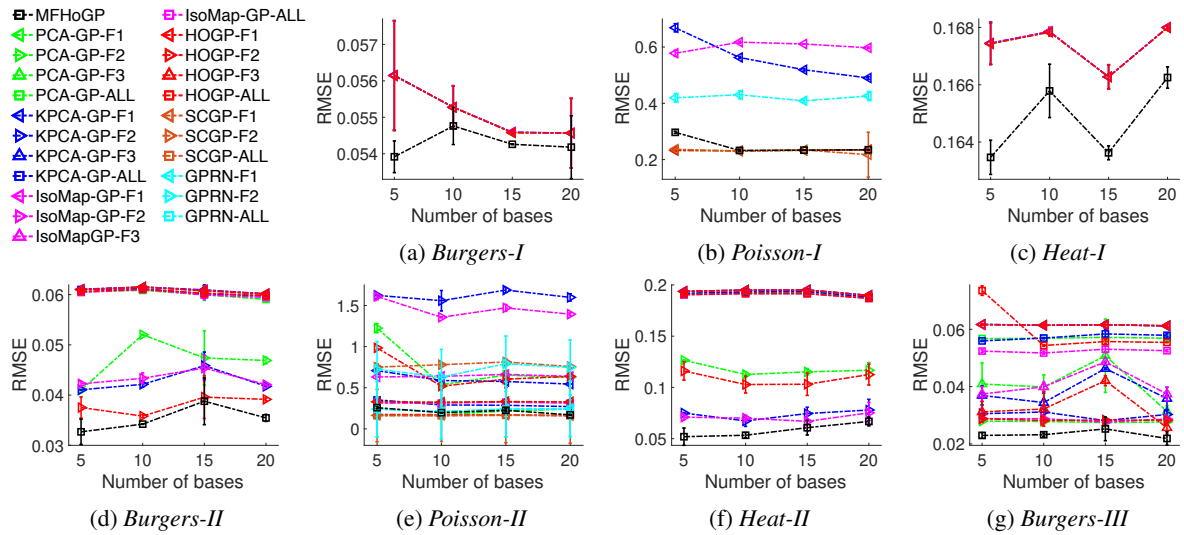
6 Experiments

6.1 Predicting Small Solution Fields

We first examined MFHoGP in predicting a relatively small number of solution outputs. These datasets were collected from solving three fundamental partial differential equations (PDEs), Burgers', Poisson's and heat equations (Olsen-Kettle, 2011) in small spatial/temporal domains. The sizes of the output fields for the three PDEs are 128×128, 32×32 and 100×100 (and so the output dimensions are $16K$, $1K$ and $10K$), respectively. In each example, the inputs are initial conditions and PDE parameters. We used numerical solvers to compute the solution field. The fidelity of the outputs are determined by the number of nodes/steps used in the solvers. The more the nodes/steps, the higher the fidelity. The details of the PDEs and data generation are provided in the supplementary material. For Burger's equation, we considered three training settings: (1) *Burgers-I*, 400 examples of fidelity-1 (the lowest fidelity), (2) *Burgers-II*, 400 fidelity-1 examples mixed with 4 fidelity-2 examples, (3) *Burgers-III*, 400 fidelity-1, 40 fidelity-2 and 4 fidelity-3 examples. Similarly, we considered two training settings for Poisson's and heat equations: (4) *Poisson-I* and (5) *Heat-I*, 400 fidelity-1 examples, (6) *Poisson-II*, 400 fidelity-1 and 10 fidelity-2 examples, and (7) *Heat-II*, 400 fidelity-1 and 4 fidelity-2 examples. For each setting, we used 112 examples with one higher fidelity for testing; we randomly sampled the input parameters and generated 5 training and test datasets. Note that the high-fidelity samples are much fewer than the low-fidelity ones; the ratio ranges from $1/100$ and $1/10$. While the output dimensions are relatively small ($\sim 10^4$), the size of training data are even smaller ($\sim 10^2$).

Competing Methods. We compared MFHoGP with three popular LMC methods/variants for scalable multi-output regression: (1) PCA-GP (Higdon et al., 2008), (2) IsoMap-GP (Xing et al., 2015),and (3) KPCA-GP (Xing et al., 2016), which obtain the bases or low-rank structures from Principal Component Analysis (PCA), IsoMap (Balasubramanian and Schwartz, 2002) and Kernel PCA (Schölkopf et al., 1998), respectively. In addition, we compared with (4) GPRN (Wilson et al., 2012), (5) SCGP, the sparse convolved GP (Alvarez and Lawrence, 2009), and (6) HOGP, high-order Gaussian process for regression (Zhe et al., 2019), a recent approach that tensorizes the outputs and can flexibly capture nonlinear output correlations and efficiently handle very high-dimensional outputs.

Parameter settings. We implemented MFHoGP with TensorFlow (Abadi et al., 2016), and used Adam (Kingma and Ba, 2014) for stochastic optimization. In the training, we set the learning rate to 10^{-3} and ran Adam for 5K epochs. For SCGP, we used the implementation from the authors' group (`https://github.com/SheffieldML/multigp`). For GPRN, we tested the efficient implementation (`https://github.com/trungngv/gprn`) from Nguyen and Bonilla (2013). We used their default settings. All the other methods were implemented with Matlab and used L-BFGS for optimization. We used RBF kernel for all the methods. For each dataset, MFHoGP integrates the examples of all the fidelities for training. Since the competing methods are developed for single-fidelity data, we conducted their training on the examples of each fidelity separately, and on all the examples



(a) *Burgers-I* (b) *Poisson-I* (c) *Heat-I*

(d) *Burgers-II* (e) *Poisson-II* (f) *Heat-II* (g) *Burgers-III*

Figure 2: The root-mean-squared-error (RMSE) of all the multi-output regression methods on small datasets in seven evaluation settings. In each setting, the results are averaged from 5 runs. After the dash in each caption (*e.g.,* "*Burgers-II*") is how many fidelities across the training data. -F{1,2,3} indicates the model trained with a particular fidelity's examples and -F-ALL with all the examples. Note that quite a few methods obtained very close results and their curves overlap (*e.g.,* in "*Heat-I*").

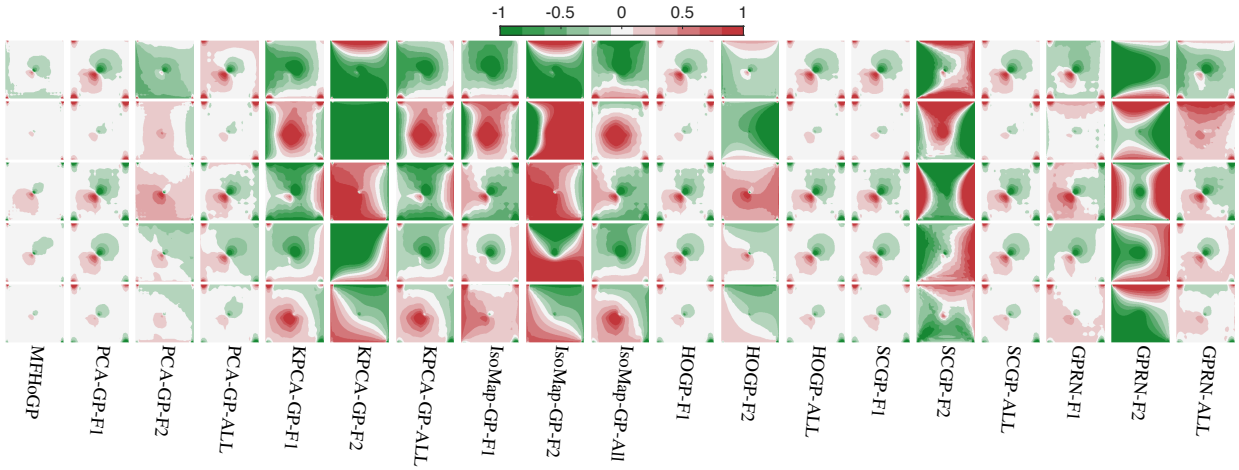
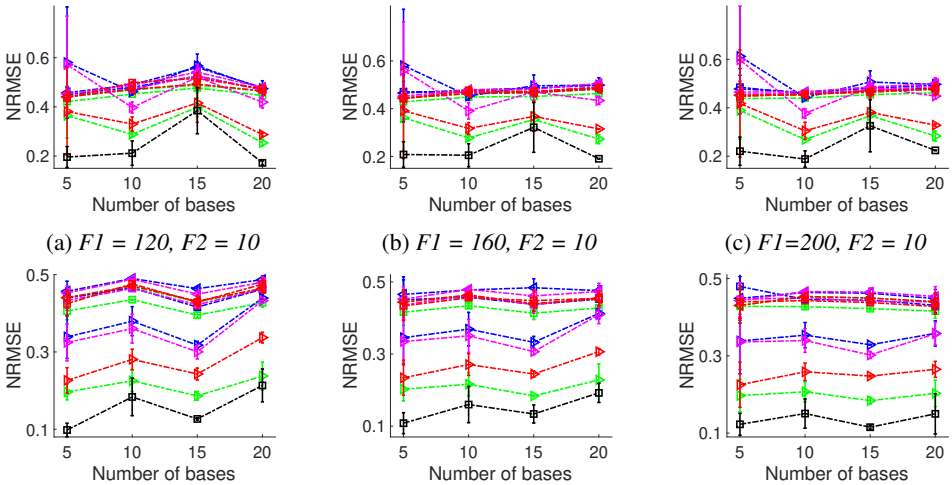


Figure 3: Visualization of local errors. Each image represents the difference between the prediction and ground-truth over individual outputs of one test example in *Poisson-II*.



(a) *F1 = 120, F2 = 10* (b) *F1 = 160, F2 = 10* (c) *F1=200, F2 = 10*

(d) *F1 = 120, F2 = 20* (e) *F1= 160, F2= 20* (f) *F1 = 200, F2 = 20*

Figure 4: Normalized RMSE in predicting 1 million dimensional pressure fields of lid-driven cavity flows. F1 and F2 are the numbers of examples with fidelity 1 and 2, respectively. In each setting, the results are averaged from 5 test sets.

merged together. For instance, -F1 denotes training with the examples of fidelity-1, -F2 with fidelity-2, and -ALL with all the examples. For overlapping inputs across fidelities (See Sec. 3.2), we preserve the higher-fidelity examples in the merged set. We varied the number of bases from $\{5, 10, 15, 20\}$, and ran all the methods on the 5 training/test datasets in each setting. For MFHoGP, we decomposed the bases according to the shapes of the output fields (see Sec. 4.1). We computed the average root-mean-square error (RMSE) and test log likelihood, and their standard deviations of all the methods. The RMSEs are reported in Fig. 2. Due to the space limit, the test log likelihoods are reported in the supplementary material. GPRN and SCGPR are only feasible for the smallest datasets *Poisson-I* and *Poisson-II* (with $\sim 1K$ outputs). For other dataset ($\geq 10K$ outputs), they either failed with excessive memory consumption, crashed or ran forever without responses. These might be due to the cost in estimating a large number of GPs and complex computation in convolution kernels.

From Fig. 2, we can see that MFHoGP obtains the smallest prediction error in almost all the cases. In many cases, MFHoGP significantly outperforms the competing approaches (p-value < 0.05, shown by the non-overlapping standard error bars (Minka, 2002)). Note that while SCGP exhibits excellent performance on *Poisson-I* and *-II*, it is inefficient and cannot deal with larger numbers of outputs, *e.g.,* over 10K. MFHoGP exhibits superior performance in terms of the test log likelihood as well (see Fig. 1 in the supplementary material). Note that for the competing methods, simply combining all the examples of different fidelities fails to achieve an improvement. In most cases, the performance is in between only training with samples of the lowest fidelity and higher ones (*e.g.,* HOGP on *Burgers-II* and *Heat-II*, PCA-GP on *Burgers-II*, *Heat-II* and *Burgers-III*). Therefore, it demonstrates the effectiveness of our approach in integrating multi-fidelity examples, even if the high fidelity samples take a tiny portion. On the single-fidelity data (see Fig. 2a-c), our model usually improves upon the LMC methods as well. It might be because the proposed nonlinear coregionalization more accurately captures the (nonlinear) output correlations and less overfits. Finally, we also examined training our model without bases decomposition: the inference is much slower and the performance is comparable or even worse. For example, in *Burgers-I* setting, bases #=15, both approaches obtain almost the same RMSE, but the bases decomposition has 3.7x speed-up.

6.2 Local Output Recovery

Next, we examined how the outputs are individually recovered, *i.e.,* how the predictive performance varies locally. To this end, we randomly selected a few test samples, and visualized the difference between the prediction and ground-truth of every single output. Fig. 3 shows the results of 5 test samples in *Poisson-II* setting. As we can see, in most regions (rendered by grey), MFHoGP achieves (almost) zero error, and only in a few small regions, it obtains small errors shown in light colors. By contrast, most competing methods result in larger errors (showed in darker colors), spreading over the vast majority of the output regions. Note that PCA-GP-F1, HOGP-F1/ALL and SCGP-F1/ALL obtained very similar local output predictions. In other settings, MFHoGP exhibits better results as well. See the supplementary material for details. Therefore, our method not only yields a superior global accuracy (as shown in Fig. 2), but locally also better recovers each individual output.

6.3 Large-Scale Flow Simulation

Finally, we applied MFHoGP in a large-scale physical simulation problem. We aimed to predict a one-million dimensional pressure field for lid-driven cavity flows (Bozeman and Dalton, 1973). When the fluid is inside a cavity and driven by a lid (or several lids) on the edge, the internal pressure can be unevenly distributed, leading to turbulent flows. Given the boundary condition, the pressure field can be determined by solving the incompressible Navier-Stokes (NS) equations (Chorin, 1968), which are known to be computationally challenging. To predict the high-dimensional field, we prepared training examples of two fidelities. We varied the number of low fidelity samples from $\{120, 160, 200\}$ and high fidelity samples from $\{10, 20\}$. For each fidelity combination, we randomly sampled the boundary conditions and simulated 5 test sets, each including 30 examples (3×10^7 outputs). The ground-truth are computed with very dense grids in finite difference. For MFHoGP, we decomposed each basis with three 100 dimensional vectors. We reported the average normalized root-mean-square error (NRMSE) and standard deviation in Fig. 4. As we can see, our method consistently improves upon the competing methods, and in many cases significantly ($p < 0.05$). Again, even combining the examples of all the fidelities, the competing methods failed to obtain improved accuracy. The results confirm the advantages of MFHoGP in learning a function with massive outputs from very limited data with different fidelities, which is common in physical simulation. The average per-epoch/-iteration time for MFHoGP, PCA-GP, KPCA-GP, IsoMap-GP and HOGP are 36.6, 11.7, 167.6, 99.1 and 3, 417.1 seconds, respectively (when the bases # is 5). Hence, MFHoGP is much faster than HOGP and has a comparable speed to the other scalable multi-output regression approaches. MFHoGP also exhibits smaller local errors (in recovering individual outputs). The local visualization results are provided in the supplementary material.

7 Conclusion

We have presented MFHoGP, a multi-fidelity high-order GP model for physical simulation. In the future, we will explore MFHoGP in other domains, such as multi-resolution large-scale sensor networks output prediction. We will further extend MFHoGP for multi-fidelity Bayesian optimization (Li et al., 2020) and active learning for complex system optimization and design problems.

Acknowledgments

This work has been supported by DARPA TRADES Award HR0011-17-2-0016, MURI-FA9550-20-1-0358, and NSF IIS-1910983.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pages 265–283.

Alvarez, M. and Lawrence, N. D. (2009). Sparse convolved gaussian processes for multi-output regression. In Advances in neural information processing systems, pages 57–64.

Álvarez, M., Luengo, D., Titsias, M., and Lawrence, N. (2010). Efficient multioutput gaussian processes through variational inducing kernels. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pages 25–32.

Alvarez, M., Ward, W., and Guarnizo, C. (2019). Non-linear process convolutions for multi-output gaussian processes. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 1969–1977.

Alvarez, M. A., Rosasco, L., Lawrence, N. D., et al. (2012). Kernels for vector-valued functions: A review. Foundations and Trends® in Machine Learning, 4(3):195–266.

Balasubramanian, M. and Schwartz, E. L. (2002). The isomap algorithm and topological stability. Science, 295(5552):7–7.

Bonilla, E. V., Agakov, F. V., and Williams, C. K. (2007). Kernel multi-task learning using task-specific features. In Artificial Intelligence and Statistics, pages 43–50.

Bonilla, E. V., Chai, K. M., and Williams, C. (2008). Multi-task gaussian process prediction. In Advances in neural information processing systems, pages 153–160.

Boyle, P. and Frean, M. (2005). Dependent gaussian processes. In Advances in neural information processing systems, pages 217–224.

Bozeman, J. D. and Dalton, C. (1973). Numerical study of viscous flow in a cavity. Journal of Computational Physics, 12(3):348–363.

Chorin, A. J. (1968). Numerical solution of the navier-stokes equations. Mathematics of computation, 22(104):745–762.

Cutajar, K., Pullin, M., Damianou, A., Lawrence, N., and González, J. (2019). Deep gaussian processes for multi-fidelity modeling. arXiv preprint arXiv:1903.07320.

Damianou, A. and Lawrence, N. (2013). Deep gaussian processes. In Artificial Intelligence and Statistics, pages 207–215.

Goulard, M. and Voltz, M. (1992). Linear coregionalization model: tools for estimation and choice of cross-variogram matrix. Mathematical Geology, 24(3):269–286.

Gupta, A. and Nagar, D. (1999). Matrix Variate Distributions, volume 104. CRC Press.

Hamelijnck, O., Damoulas, T., Wang, K., and Girolami, M. (2019). Multi-resolution multi-task gaussian processes. arXiv preprint arXiv:1906.08344.

Harshman, R. A. (1970). Foundations of the PARAFAC procedure: Model and conditions for an"explanatory"multi-mode factor analysis. UCLA Working Papers in Phonetics, 16:1–84.

Hebbal, A., Brevault, L., Balesdent, M., Talbi, E.-G., and Melab, N. (2019). Multi-fidelity modeling using DGPs: Improvements and a generalization to varying input space dimensions. In NeurIPS Workshop on Bayesian Deep Learning.

Higdon, D. (2002). Space and space-time modeling using process convolutions. In Quantitative methods for current environmental issues, pages 37–56. Springer.

Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008). Computer model calibration using high-dimensional output. Journal of the American Statistical Association, 103(482):570–583.

Journel, A. G. and Huijbregts, C. J. (1978). Mining geostatistics, volume 600. Academic press London.

Keane, A. J. and Nair, P. B. (2005). Computational approaches for aerospace design. John Wiley&Sons, Ltd, West Sussex, 582.

Kennedy, M. C. and O'Hagan, A. (2000). Predicting the output from a complex computer code when fast approximations are available. Biometrika, 87(1):1–13.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

Li, S., Xing, W., Kirby, R., and Zhe, S. (2020). Multi-fidelity Bayesian optimization via deep neural networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, Advances in Neural Information Processing Systems, volume 33, pages 8521–8531. Curran Associates, Inc.

Matheron, G. (1982). Pour une analyse krigeante des données régionalisées. Centre de Géostatistique, Report N-732, Fontainebleau.

Minka, T. P. (2002). Judging significance from error bars. Technical report, MIT.

Nguyen, T. and Bonilla, E. (2013). Efficient variational inference for gaussian process regression networks. In Artificial Intelligence and Statistics, pages 472–480.

Oakley, J. and O'Hagan, A. (2002). Bayesian inference for the uncertainty distribution of computer model outputs. Biometrika, 89(4):769–784.

Olsen-Kettle, L. (2011). Numerical solution of partial differential equations. Lecture notes at University of Queensland, Australia.

Peherstorfer, B., Willcox, K., and Gunzburger, M. (2018). Survey of multifidelity methods in uncertainty propagation, inference, and optimization. Siam Review, 60(3):550–591.

Peiró, J. and Sherwin, S. (2005). Finite difference, finite element and finite volume methods for partial differential equations. In Handbook of materials modeling, pages 2415–2446. Springer.

Perdikaris, P., Raissi, M., Damianou, A., Lawrence, N., and Karniadakis, G. E. (2017). Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 473(2198):20160751.

Raissi, M. (2018). Deep hidden physics models: Deep learning of nonlinear partial differential equations. The Journal of Machine Learning Research, 19(1):932–955.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378:686–707.

Rakitsch, B., Lippert, C., Borgwardt, K., and Stegle, O. (2013). It is all in the noise: Efficient multi-task gaussian process inference with structured residuals. In Advances in neural information processing systems, pages 1466–1474.

Rasmussen, C. E. and Ghahramani, Z. (2002). Infinite mixtures of gaussian process experts. In Advances in neural information processing systems, pages 881–888.

Santner, T. J., Williams, B. J., Notz, W., and Williams, B. J. (2003). The design and analysis of computer experiments, volume 1. Springer.

Schölkopf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. Neural computation, 10(5):1299–1319.

Stegle, O., Lippert, C., Mooij, J. M., Lawrence, N. D., and Borgwardt, K. (2011). Efficient inference in matrix-variate gaussian models with\iid observation noise. In Advances in neural information processing systems, pages 630–638.

Wilson, A. G., Knowles, D. A., and Ghahramani, Z. (2012). Gaussian process regression networks. In Proceedings of the 29th International Coference on International Conference on Machine Learning, pages 1139–1146. Omnipress.

Xing, W., Shah, A. A., and Nair, P. B. (2015). Reduced dimensional gaussian process emulators of parametrized partial differential equations based on isomap. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, volume 471, page 20140697. The Royal Society.

Xing, W., Triantafyllidis, V., Shah, A., Nair, P., and Zabaras, N. (2016). Manifold learning for the emulation of spatial fields from computational models. Journal of Computational Physics, 326:666–690.

Zhe, S., Xing, W., and Kirby, R. M. (2019). Scalable high-order gaussian process regression. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 2611–2620.

Zienkiewicz, O. C., Taylor, R. L., Zienkiewicz, O. C., and Taylor, R. L. (1977). The finite element method, volume 36. McGraw-hill London.