GAR: Generalized Autoregression for Multi-Fidelity Fusion

Anonymous Author(s)
Affiliation
Address
email

Abstract

In many scientific research and engineering applications, where repeated simulations of complex systems are conducted, a surrogate is commonly adopted to quickly estimate the whole system. To reduce the expensive cost of generating training examples, it has become a promising approach to combine the results of low-fidelity (fast but inaccurate) and high-fidelity (slow but accurate) simulations. Despite the fast developments of multi-fidelity fusion techniques, most existing methods require particular data structures and do not scale well to high-dimensional output. To resolve these issues, we generalize the classic autoregression (AR), which is wildly used due to its simplicity, robustness, accuracy, and tractability, and propose generalized autoregression (GAR) using tensor formulation and latent features. GAR can deal with arbitrary dimensional outputs and arbitrary multi-fidelity data structure to satisfy the demand of multi-fidelity fusion for complex problems; it admits a fully tractable likelihood and posterior requiring no approximate inference and scales well to high-dimensional problems. Furthermore, we prove the autokrigeability theorem based on GAR in the multi-fidelity case and develop CIGAR, a simplified GAR with the exact predictive mean accuracy with computation reduction by a factor of d^3, where d is the dimensionality of the output. In experiments of canonical PDEs and scientific computational examples, the proposed method consistently outperforms the SOTA methods with a large margin (up to 6x improvement in RMSE) with only a couple high-fidelity training samples.

1 Introduction

The design, optimization, and control of many systems in science and engineering can rely heavily on the numerical analysis of differential equations, which is generally computationally intense. In this case, a data-driven surrogate model is used to approximate the system based on the input-output data of the numerical solver and to help improve convergence efficiency where repeated simulations are required, e.g., in Bayesian optimization (BO) [1] and uncertainty analysis (UQ) [2].

With the surrogate model in place, the remaining challenge is that executing the high-fidelity numerical solver to generate training data can still be very expensive. To further reduce the computational burden, it is possible to combine low-fidelity results to make high-fidelity predictions [3]. More specifically, low-fidelity solvers are normally based on simplified PDEs (e.g., reducing the levels of physical detail) or simple solver setup (e.g., using a coarse mesh, a large time step, a lower order of an approximating basis, and a higher error tolerance). They provide fast but inaccurate solutions to the original problems where as the high-fidelity solver is accurate yet expensive. The multi-fidelity fusion technique works similar to transfer learning to utilize many low-fidelity observations to improve the accuracy when using only a few high-fidelity samples. In general, it involves constructions of surrogates for different fidelities and a cross-fidelity transition process. Due to its efficiency,

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

multi-fidelity has attracted increasing attention in BO [4, 5], UQ [6, 7], and surrogate modeling [8, 9]l. We refer to [10, 11] for a recent review.

Despite the success of many state-of-the-art (SOTA) approaches, they normally assume that (1) the output dimension is the same and aligned across all fidelities, which generally does not hold for multi-fidelity simulation where the output are quantities at nodes that are not aligned; (2) the high-fidelity samples' corresponding inputs form a subset of the low-fidelity inputs; and (3) the output dimension is small, which in not practical for scientific computing where dimension can be 1-million (for a $100 \times 100 \times 100$ spatial-temporal field). These assumptions seriously hinder their applications for modern high-dimensional problems, which is common in scientific computing for solving PDEs and real-world datasets, e.g., MRI imaging.

To resolve these challenges, previous work either uses interpolation to align the dimension [12, 9] or relies on approximate inference with brutal simplification [8, 13], leading to inferior performance. We notice that the classic autoregression (AR), which is popularly used due to its simplicity, robustness, accuracy, and tractability, consistently shows robust and top-tier performance for different datasets in the literature, despite its incapability for high-dimensional problems. Thus, instead of proposing another ad-hoc model with pre-processing and simplification (leading to models that are difficult to tune and generalize poorly), we generalize AR with tensor algebra and latent features and propose generalized autoregression (GAR), which can deal with arbitrary high-dimensional Problems without the subset multi-fidelity data structure. GAR is a fully tractable Bayesian model with scalability to extremely high-dimensional outputs, without requiring any approximate inference. The novelty of this work is as follows,

1. Generalization of the AR for arbitrary non-structured high-dimensional outputs. With tensor algebra and latent features, GAR allows effective knowledge transfer in closed-form and is scalable to extreme high-dimensional problems.
2. Generalization to non-subset multi-fidelity data. To the best of our knowledge, we are the first to generalize the closed form solution of subset data to non-subset cases for the AR and the proposed GAR.
3. For the first time, we reveal the autokrigeability [14] for the multi-fidelity fusion within an AR structure, based on which we derive conditional independent GAR (CIGAR), an efficient implementation of GAR with the exact accuracy in posterior mean predictions.

2 Backgronud

2.1 Statement of the problem

Given multi-fidelity data $D^i = \{\mathbf{x}_n^i, y_n^i\}_{n=1}^{N^i}$, for $i = 1, \ldots, \tau$, where $\mathbf{x} \in \mathbb{R}^l$ denotes the system inputs (a vector of parameters that appear in the system of equations and/or in the initial-boundary conditions for a simulation); $\mathbf{y}^i \in \mathbb{R}^{d^i}$ denotes the corresponding outputs, where d^i is the dimension for i fidelity; τ is the total number of fidelities. Generally speaking, higher-fidelity are closer to the ground truth and are expensive to obtain, and thus we have fewer samples for high fidelity, i.e., $N^1 > N^2 > \cdots > N^\tau$. The dimensionality d^i is not necessary the same or aligned across different fidelities. In most work e.g., [15, 16, 11], the system inputs of higher-fidelity are chosen to be the subset of the lower-fidelity, i.e., $\mathbf{X}^\tau \subset \cdots \subset \mathbf{X}^2 \subset \mathbf{X}^1$. We call this the subset structure for a multi-fidelity dataset, as opposed to arbitrary data structures, which we will resolve in Section section 3.3 with a closed-form solution and extend it to the classic AR. Our goal is to estimate the function $\mathbf{f}^\tau : \mathbb{R}^l \to \mathbb{R}^{d^\tau}$ given the observation data across different fidelities $\{D^i\}_{i=1}^\tau$.

2.2 Autoregression

For the sake of clarity, we consider a two-fidelity case with superscript h indicating high-fidelity and l low-fidelity. Nevertheless, the formulation can be generalized to problems with multiple fidelities recursively. Considering a simple scalar output for all fidelities, the AR [3] assumes

$$f^h(\mathbf{x}) = \rho f^l(\mathbf{x}) + f^r(\mathbf{x}), \tag{1}$$

where ρ is a factor transferring knowledge from the low fidelity in a linear fashion whereas $f^r(\mathbf{x})$ tries to capture the residual information. If we assume a zero mean Gaussian process (GP) prior

[17] (see Appendix A for a brief description) for $f^l(\mathbf{x})$ and $f^r(\mathbf{x})$, i.e., $f^l(\mathbf{x}) \sim \mathcal{N}(0, k^l(\mathbf{x}, \mathbf{x}'))$ and $f^r(\mathbf{x}) \sim \mathcal{N}(0, k^r(\mathbf{x}, \mathbf{x}'))$, the high-fidelity function also follows a GP. This gives an elegant joint GP for the joint observations $\mathbf{y} = [\mathbf{y}^l; \mathbf{y}^h]^T$,

$$\begin{pmatrix} \mathbf{y}^l \\ \mathbf{y}^h \end{pmatrix} \sim \mathcal{N}\begin{pmatrix} \mathbf{0}, & \mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^l) & \rho\mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^h) \\ & \rho\mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^l) & \rho^2\mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^h) + \mathbf{K}^r(\mathbf{X}^h, \mathbf{X}^h) \end{pmatrix} \tag{2}$$

where $\mathbf{y}^l \in \mathbb{R}^{N_l \times 1}$ is the low-fidelity observations corresponding to input $\mathbf{X}^l \in \mathbb{R}^{N_l \times L}$ and $\mathbf{y}_h \in \mathbb{R}^{N_h \times 1}$ is the high-fidelity observations; $[\mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^l)]_{ij} = k^l(\mathbf{x}_i, \mathbf{x}_j)$ is the covariance matrix of the low-fidelity inputs $\mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}^l$; $[\mathbf{K}^r(\mathbf{X}^l, \mathbf{X}^l)]_{ij} = k^r(\mathbf{x}_i, \mathbf{x}_j)$ is for the high-fidelity inputs $\mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}^h$; $[\mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^h)]_{ij} = k^l(\mathbf{x}_i, \mathbf{x}_j)$ is the cross-fidelity covariance matrix of the low-fidelity inputs $\mathbf{x}_i \in \mathbf{X}^l$ and high-fidelity inputs $\mathbf{x}_j \in \mathbf{X}^h$, and $\mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^l) = (\mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^h))^T$. One immediate advantage of the AR is that the joint Gaussian form allows not only joint training for all low- and high-fidelity data but also predictions for any given new inputs by conditional Gaussian (as the posterior is derived in a standard GP [17]). Furthermore, Le Gratiet [15] derive Lemma 1 to reduce the complexity from $O((N^l + N^h)^3)$ to $O((N^l)^3 + (N^h)^3)$ with a subset data structure.

Lemma 1. *[15] If $\mathbf{X}^h \subset \mathbf{X}^l$, the joint likelihood and predictive posterior of the AR can be decomposed into two independent parts corresponding to the low- and high-fidelity data.*

3 Generalized Autoregression

Let's now consider the high-dimensional case. A naive approach is to simply convert the multi-dimensional output into a scalar value by attaching a dimension index to the input. However, the AR will end up with a joint GP with a covariance matrix of the size of $(N^l d^l + N^h d^h)^2$, making it infeasible for large dimensional problems.

3.1 Tensor Factorized Generalization with Latent Features

To resolve the scalability issue, we rearrange all the output into a multidimensional space (i.e., a tensor space) and introduce latent coordinate features to index the outputs to capture their correlations as in HOGP [18]. More specifically, we organize the low-fidelity outputs as a M-mode tensor, $\mathbf{Z}^l \in \mathbb{R}^{d_1^l \times \cdots \times d_M^l}$, where the output dimension $d^l = \prod_{m=1}^M d_m^l$. The element $\mathcal{Z}^l$ is indexed based on its coordinates $\mathbf{c} = (c_1, \ldots, c_M)(1 \leqslant c_k \leqslant d_k$ for $k = 1, \ldots, M)$. If the original data indeed admit a multi-array structure, we can use their original index with actual meaning to index the coordinates. For instance, a 2D spatial dataset can use its original spatial coordinate to index a single location (pixel). To improve our model flexibility, we do not have to limit ourselves from using the original index, particularly for the cases where the original data does not admit a multi-array structure or the multi-array structure is of too large size. In such case, we can use arbitrary tensorization and a latent feature vector $\mathbf{v}_{c_m}^l$ (whose values are inferred in model training) for each coordinate c_m in mode m. This way, the element $\mathcal{Z}^l$ is indexed by the vector $(\mathbf{v}_{c_1}^l, \ldots, \mathbf{v}_{c_M}^l)$. Following the linear transformation of Eq. (1), we first introduce the tensor-matrix product [19] at mode m,

$$\mathbf{F}^h(\mathbf{x}) = \mathbf{F}^l(\mathbf{x}) \times_1 \mathbf{W}_1, \ldots, \times_M \mathbf{W}_M + \mathbf{F}^r(\mathbf{x}), \tag{3}$$

where $\mathbf{F}^h(\mathbf{x})$ denotes target function $\mathbf{f}^h(\mathbf{x})$ with its output organized into multi-array $\mathbf{Z}^h$, and the same concept applies to $\mathbf{F}^l(\mathbf{x})$ and $\mathbf{F}^r(\mathbf{x})$; $\times_m$ denotes the tensor-matrix product at mode m. To give a concrete example, considering an arbitrary tensor $\mathbf{Z}^l \in \mathbb{R}^{d_1^l \times \cdots \times d_M^l}$ and a matrix $\mathbf{W}_m \in \mathbb{R}^{s \times d_m}$, the $\times_m$ product is calculated as $[\mathbf{Z}^l \times_m \mathbf{W}_m]_{i_1, \ldots, i_{m-1}, j, i_{m+1} \ldots, i_M} = \sum_{k=1}^{c_m} w_{jk} \mathcal{Z}_{i_1, \ldots, i_{m-1}, k, i_{m+1} \ldots, i_M}$, which becomes an $d_1^l \times \cdots \times d_{m-1}^l \times s \times d_{m+1}^l \times \cdots \times d_M^l$ tensor. We can further denote the group of M linear transformation matrixes as a Tucker tensor $\mathbf{W} = [\mathbf{W}_1, \ldots, \mathbf{W}_M]$ and represent Eq. (3) compactly using a Tucker operator [19], $\mathbf{F}^l(\mathbf{x}) \times \mathbf{W}$, which has an important property:

$$\operatorname{vec}\left(\mathbf{F}^h(\mathbf{x}) - \mathbf{F}^r(\mathbf{x})\right) = (\mathbf{W}_1 \otimes \ldots \otimes \mathbf{W}_M) \operatorname{vec}\left(\mathbf{F}^l(\mathbf{x})\right). \tag{4}$$

Inspired by the AR of Eq. (1), we place a tensor-variate GP (TGP) prior [20] for the low-fidelity tensor function $\mathbf{F}^l(\mathbf{x})$ and the residual tensor function $\mathbf{F}^r(\mathbf{x})$:

$$\mathbf{Z}^l(\mathbf{x}, \mathbf{x}') \sim \mathcal{TGP}\left(\mathbf{0}, k^l(\mathbf{x}, \mathbf{x}'), \mathbf{S}_1^l, \ldots, \mathbf{S}_M^l\right), \mathbf{Z}^r(\mathbf{x}, \mathbf{x}') \sim \mathcal{TGP}\left(\mathbf{0}, k^r(\mathbf{x}, \mathbf{x}'), \mathbf{S}_1^r, \ldots, \mathbf{S}_M^r\right), \tag{5}$$

where $\mathbf{S}_m^i \in \mathbb{R}^{d_m \times d_m}$ are the output correlation matrix with $[\mathbf{S}_m^i]_{jk} = \tilde{k}_m^i(\mathbf{v}_{c_i}^i, \mathbf{v}_{c_k}^i)$ and $\tilde{k}_m^i(\cdot,\cdot)$ being the kernel function (with unknown hyperparameters). A TGP is a generalization of a multivariate GP that essentially represents a joint GP prior $\mathrm{vec}(\mathbf{Y}^l) \sim \mathcal{N}\left(0, \mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^l) \bigotimes_{m=1}^M \mathbf{S}_m\right)$. Similar to the joint probability of (2), we can derive the joint probability for $\mathbf{y} = [\mathrm{vec}(\mathbf{Y}^l)^T, \mathrm{vec}(\mathbf{Y}^h)^T]^T$ based on Tucker transformation of (3); we keep the the proof in the Appendix for clarity.

Lemma 2. *Given the tensor GP priors for $\mathbf{Y}^l(\mathbf{x}, \mathbf{x}')$ and $\mathbf{Y}^r(\mathbf{x}, \mathbf{x}')$ and the Tucker transformation of (3), the joint probability for $\mathbf{y} = [\mathrm{vec}(\mathbf{Y}^l)^T, \mathrm{vec}(\mathbf{Y}^h)^T]^T$ is $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma} =$*

$$\begin{pmatrix} \mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^l) \otimes \left(\bigotimes_{m=1}^M \mathbf{S}_m\right) & \mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^h) \otimes \left(\bigotimes_{m=1}^M \mathbf{S}_m \mathbf{W}_m^T\right) \\ \mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^l) \otimes \left(\bigotimes_{m=1}^M \mathbf{W}_m \mathbf{S}_m\right) & \mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^h) \otimes \left(\bigotimes_{m=1}^M \mathbf{W}_m \mathbf{S}_m \mathbf{W}_m^T\right) + \mathbf{K}^r(\mathbf{X}^h, \mathbf{X}^h) \otimes \left(\bigotimes_{m=1}^M \mathbf{S}_m^r\right) \end{pmatrix}.$$

Lemma 2 admits any arbitrary outputs (living in different space, having different dimensions and/or mode, and being unaligned) at different fidelity. Also, it does not require a subset dataset to hold.

Corollary 3.0.1. *Lemma 2 can be applied to data with a different number of mode at each fidelity, i.e., $M^l \neq M^h$, if we pad the output with fewer modes with a mode having only one latent index such that all outputs have the same M number of modes.*

Lemma 2 defines our GAR model, a generalized AR with special tensor structures. The covariance for low-fidelity is $\mathrm{cov}(\mathcal{Z}_{\mathbf{c}}^l(\mathbf{x}), \mathcal{Z}_{\mathbf{c}'}^l(\mathbf{x}')) = k^l(\mathbf{x}, \mathbf{x}') \prod_{m=1}^M \tilde{k}_m^l(\mathbf{v}_{c_m}^m, \mathbf{v}_{c'_m}^m)$, cross-fidelity $\mathrm{cov}(\mathcal{Z}_{\mathbf{c}}^l(\mathbf{x}), \mathcal{Z}_{\mathbf{c}'}^h(\mathbf{x}')) = k^l(\mathbf{x}, \mathbf{x}') \prod_{m=1}^M \tilde{k}_m^l(\mathbf{v}_{c_m}^l, \mathbf{v}_{c'_m}^l) w_{c,c'}^m$ (where $w_{c,c'}^m$ is the c, c'-th element of $\mathbf{W}^m$), and high-fidelity $\mathrm{cov}(\mathcal{Z}_{\mathbf{c}}^h(\mathbf{x}), \mathcal{Z}_{\mathbf{c}'}^h(\mathbf{x}')) = k^l(\mathbf{x}, \mathbf{x}') \prod_{m=1}^M \tilde{k}_m^l(\mathbf{v}_{c_m}^l, \mathbf{v}_{c'_m}^l)(w_{c,c'}^m)^2 + k^h(\mathbf{x}, \mathbf{x}') \prod_{m=1}^M \tilde{k}_m^r(\mathbf{u}_{c_m}^m, \mathbf{u}_{c'_m}^m)(w_{c,c'}^m)^2$. The complex between-fidelity output correlations are captured using latent features $\{\mathbf{V}^m, \mathbf{U}^m\}_{m=1}^M$ with arbitrary kernel function $\tilde{k}_m^i$ whereas the cross-fidelity output correlations are captured in a composite manner. This combination overcomes the simple linear correlations assumed in previous work that simply decomposes the output as a dimension reduction preprocess [12]. When the dimensionality aligns for $\mathbf{Z}^l$ and $\mathbf{Z}^h$ and thus $d_m^l = d_m^h$, we can share the same latent features across the two fidelity data by letting $\mathbf{v}_j^m = \mathbf{u}_j^m$ while keeping the kernel functions different. This way, the latent features are more resistant to overfitting. For non-aligned data with explicit indexing, we can use kernel interpolation [21] for the same purpose. To further encourage sparsity in the latent feature, we impose a Laplace prior, i.e., $\mathbf{v}_j^m \sim \mathrm{Laplace}(\lambda) \propto \exp(-\lambda ||\mathbf{v}_j^m||_1)$.

3.2 Efficient Model Inference for Subset Data Structure

With the model fully defined, we can now train the model to obtain all unknown model parameters. For compactness, we use the following compact notation $\mathbf{S}^l = \bigotimes_{m=1}^M \mathbf{S}_m^l$, $\mathbf{S}^h = \bigotimes_{m=1}^M \mathbf{S}_m^h$, $\mathbf{W} = \bigotimes_{m=1}^M \mathbf{W}_m$, $\mathbf{K}^l = \mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^l)$, $\mathbf{K}^{lh} = \mathbf{K}^l(\mathbf{X}^l, \mathbf{X}^h)$, $\mathbf{K}^{hl} = \mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^l)$, $\mathbf{K}^{lr} = \mathbf{K}^l(\mathbf{X}^h, \mathbf{X}^h)$, and, $\mathbf{K}^r = \mathbf{K}^r(\mathbf{X}^h, \mathbf{X}^h)$ (with a slight abuse of notation).

Lemma 3. *Tensor generalization of Lemma 1. If $\mathbf{X}^h \subset \mathbf{X}^l$, the joint likelihood $\mathcal{L}$ for $\mathbf{y} = [\mathrm{vec}(\mathbf{Y}^l)^T, \mathrm{vec}(\mathbf{Y}^h)^T]^T$ admits two independent separable likelihoods $\mathcal{L} = \mathcal{L}^l + \mathcal{L}^r$, where*

$$\mathcal{L}^l = -\frac{1}{2}\mathrm{vec}\left(\mathbf{Y}^l\right)^T (\mathbf{K}^l \otimes \mathbf{S}^l)^{-1} \mathrm{vec}\left(\mathbf{Y}^l\right) - \frac{1}{2}\log|\mathbf{K}^l \otimes \mathbf{S}^l| - \frac{N^l D^l}{2}\log(2\pi),$$

$$\mathcal{L}^r = -\frac{1}{2}\mathrm{vec}\left(\mathbf{Y}^h - \mathbf{Y}^l \times \hat{\mathbf{W}}\right)^T (\mathbf{K}^r \otimes \mathbf{S}^r)^{-1} \mathrm{vec}\left(\mathbf{Y}^h - \mathbf{Y}^l \times \hat{\mathbf{W}}\right) - \frac{1}{2}\log|\mathbf{K}^r \otimes \mathbf{S}^r| - \frac{N^h D^h}{2}\log(2\pi),$$

where $\hat{\mathbf{W}} = [\mathbf{E}, \mathbf{W}_1, \ldots, \mathbf{W}_M]$ is a Tucker tensor with selection matrix $\mathbf{E}^T \mathbf{X}^l = \mathbf{X}^h$.

We preserve the proof in Appendix for clarity. Note that $\mathcal{L}^l$ and $\mathcal{L}^r$ are HOGP likelihoods for $\mathbf{Y}^l$ and residual $\mathbf{Y}^h - \mathbf{Y}^l \times \hat{\mathbf{W}}$, respectively. Since the computational of $\mathcal{L}^l$ and $\mathcal{L}^r$ are independent, the model training can be conducted efficiently in parallel.

Predictive posterior. Similarly, we can derive the concrete predictive posterior for high-fidelity outputs by integrating out the latent functions after some tedious linear algebra (see Appendix), which

is also Gaussian, $\mathrm{vec}(\mathbf{Z}_*^h) \sim \mathcal{N}(\mathrm{vec}(\bar{\mathbf{Z}}_*^h), \mathbf{S}_*^h)$, where

$$\begin{aligned}\mathrm{vec}(\bar{\mathbf{Z}}_*^h) &= \left(\mathbf{k}_*^l \left(\mathbf{K}^l\right)^{-1} \otimes \mathbf{W}\right) \mathrm{vec}(\mathbf{Y}^l) + \left(\mathbf{k}_*^r \left(\mathbf{K}^r\right)^{-1} \otimes \mathbf{I}\right) \mathrm{vec}(\mathbf{Y}^r), \\ \mathbf{S}_*^h &= \left(k_{**}^l - (\mathbf{k}_*^l)^T \left(\mathbf{K}^l\right)^{-1} \mathbf{k}_*^l\right) \otimes \mathbf{W}\mathbf{S}^l\mathbf{W}^T + \left(k_{**}^r - (\mathbf{k}_*^r)^T \left(\mathbf{K}^r\right)^{-1} \mathbf{k}_*^r\right) \otimes \mathbf{S}^r,\end{aligned} \tag{6}$$

where $\mathbf{k}_*^l = \mathbf{k}^l(\mathbf{x}_*, \mathbf{X}^l)$ is the vector of covariance between the give input $\mathbf{x}_*$ and low-fidelity observation inputs $\mathbf{X}^l$ and similarly, $\mathbf{k}_{**}^l = \mathbf{k}^l(\mathbf{x}_*, \mathbf{x}_*)$, $\mathbf{k}_*^r = \mathbf{k}^r(\mathbf{x}_*, \mathbf{X}^h)$, $\mathbf{k}_{**}^r = \mathbf{k}^r(\mathbf{x}_*, \mathbf{x}_*)$.

3.3 Generalization for Non-subset Data: Efficient Model Inference and Prediction

In practice, it is sometimes difficult to ask the multi-fidelity data to preserve a subset structure, particularly in the case of multi-fidelity Bayesian optimization [22, 23]. This presents the challenge for most SOTA multi-fidelity models e.g., NAR [16], ResGP [9], stochastic collocation [24]. In contrast, the advantage of AR is that even if the multi-fidelity data does not admit a subset data structure, the model can still be trained using all available data based on the joint likelihood of (2). However, this method lacks scalability due to the inversion of the large joint covariance matrix $\mathbf{\Sigma}$. The situation will get worse if we are dealing with multi-fidelity data with more than two fidelities. To resolve this issue, we propose a fast inference method based on imaginary subset. More specifically, considering the missing low-fidelity data as latent variables $\hat{\mathbf{Y}}^l$, the joint likelihood function is

$$\begin{aligned}\log p(\mathbf{Y}^l, \mathbf{Y}^h) &= \log \int p(\mathbf{Y}^l, \mathbf{Y}^h, \hat{\mathbf{Y}}^l) d\hat{\mathbf{Y}}^l = \log \int \left(p(\mathbf{Y}^h|\hat{\mathbf{Y}}^l, \mathbf{Y}^l) p(\hat{\mathbf{Y}}^l|\mathbf{Y}^l) p(\mathbf{Y}^l)\right) d\hat{\mathbf{Y}}^l \\ &= \log \int p(\mathbf{Y}^h|\hat{\mathbf{Y}}^l, \mathbf{Y}^l) p(\hat{\mathbf{Y}}^l|\mathbf{Y}^l) d\hat{\mathbf{Y}}^l + \log p(\mathbf{Y}^l),\end{aligned} \tag{7}$$

where $p(\mathbf{Y}^h|\hat{\mathbf{Y}}^l, \mathbf{Y}^l)$ is the likelihood in Lemma 3 given the complementary imaginary subset, and $p(\hat{\mathbf{Y}}^l|\mathbf{Y}^l) \sim \mathcal{N}(\bar{\mathbf{Y}}^l, \hat{\mathbf{S}}^l \otimes \mathbf{S}^l)$ is the imaginary posterior with the given low-fidelity observations $\mathbf{Y}^l$. The integral can be calculated using Gaussian quadrature or other sampling methods as in [8, 25], which is slow and inaccurate.

Lemma 4. *The joint likelihood of GAR for non-subset (and also unaligned) data can be decomposed into two independent GPs'*

$$\begin{aligned}&\log p(\mathbf{Y}^l, \mathbf{Y}^h) = \mathcal{L}^l - \frac{N^h d^h}{2} \log(2\pi) - \frac{1}{2} \log \left|\mathbf{K}^r \otimes \mathbf{S}^r + \hat{\mathbf{E}}\hat{\mathbf{S}}^l\hat{\mathbf{E}}^T \otimes \mathbf{W}^T\mathbf{S}^l\mathbf{W}\right| \\ &- \frac{1}{2}\left[\begin{pmatrix} \mathrm{vec}(\check{\mathbf{Y}}^h) \\ \mathrm{vec}(\hat{\mathbf{Y}}^h) \end{pmatrix}^T - \begin{pmatrix} \mathrm{vec}(\check{\mathbf{Y}}^l) \\ \mathrm{vec}(\bar{\mathbf{Y}}^l) \end{pmatrix}^T \tilde{\mathbf{W}}^T\right] \left(\mathbf{K}^r \otimes \mathbf{S}^r + \hat{\mathbf{E}}\hat{\mathbf{S}}^l\hat{\mathbf{E}}^T \otimes \mathbf{W}^T\mathbf{S}^l\mathbf{W}\right)^{-1} \left[\begin{pmatrix} \mathrm{vec}(\check{\mathbf{Y}}^h) \\ \mathrm{vec}(\hat{\mathbf{Y}}^h) \end{pmatrix} - \tilde{\mathbf{W}} \begin{pmatrix} \mathrm{vec}(\check{\mathbf{Y}}^l) \\ \mathrm{vec}(\bar{\mathbf{Y}}^l) \end{pmatrix}\right]\end{aligned} \tag{8}$$

where $\mathcal{L}^l$ is the likelihood for low-fidelity data $\mathbf{Y}^l$, $\tilde{\mathbf{W}} = \mathbf{I}_{N^h} \otimes \mathbf{W}$ $\hat{\mathbf{Y}}^h$ is the collection of high-fidelity observations corresponding to the imaginary low-fidelity outputs $\hat{\mathbf{Y}}^l$; $\check{\mathbf{Y}}^h$ is the complement (with selection matrix $\check{\mathbf{X}}^h = \mathbf{E}^T\mathbf{X}^l$) corresponding to low-fidelity outputs $\check{\mathbf{Y}}^l$, i.e., $\mathbf{Y}^h = [\check{\mathbf{Y}}^h, \hat{\mathbf{Y}}^h]$; and $\hat{\mathbf{X}}^h = \hat{\mathbf{E}}^T\mathbf{X}^h$ are the selection matrix for $\hat{\mathbf{Y}}^l$.

We preserve the proof in the appendix. Notice that $\hat{\mathbf{E}}\hat{\mathbf{S}}^l\hat{\mathbf{E}}^T \otimes \mathbf{W}^T\mathbf{S}^l\mathbf{W} = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{S}}^l \end{pmatrix} \otimes \mathbf{W}^T\mathbf{S}^l\mathbf{W}$ is the low-right block of the predictive variance for the missing low-fidelity observations $\mathrm{vec}(\hat{\mathbf{Y}})$; We can easily understand the last part of the likelihood as a GP with accumulated uncertainty(variance) added to the corresponding missing points. Lemma 4 naturally applied to the AR by when the outputs is a scaler, where $\mathbf{W} = \rho$, $\mathbf{S}^l = 1$, and $\mathbf{S}^r = 1$

Predictive posterior. Surprisingly, the posterior also turns out to be a Gaussian distribution,

$$\begin{aligned}p\left(\mathbf{Z}_*^h|\mathbf{Y}^l, \mathbf{Y}^h, \mathbf{x}_*\right) &= 2\pi^{-\frac{d^h}{2}} \times \left|\mathbf{S}_*^h + \mathbf{\Gamma}\left(\hat{\mathbf{S}}^l \otimes \mathbf{S}^l\right)\mathbf{\Gamma}^T\right|^{-\frac{1}{2}} \\ &\times \exp\left[-\frac{1}{2}\left(\mathrm{vec}(\mathbf{Z}_*^h) - \mathrm{vec}(\bar{\mathbf{Z}})\right)^T \left(\mathbf{S}_*^h + \mathbf{\Gamma}\left(\hat{\mathbf{S}}^l \otimes \mathbf{S}^l\right)\mathbf{\Gamma}^T\right)^{-1} \left(\mathrm{vec}(\mathbf{Z}_*^h) - \mathrm{vec}(\bar{\mathbf{Z}})\right)\right],\end{aligned} \tag{9}$$

where and $\mathbf{\Gamma}$ and the mean of the predictive posterior $\bar{\mathbf{Z}}_*$ are given as follows,

$$\mathbf{\Gamma} = \left([\mathbf{k}_*^r(\mathbf{K}^r)^{-1}\mathbf{E}_n^T - \mathbf{k}_*^l(\hat{\mathbf{K}}^l)^{-1}] \otimes \mathbf{W}\right)\mathbf{E}_m \otimes \mathbf{I}^l,$$
$$\mathrm{vec}(\bar{\mathbf{Z}}_*) = \left(\mathbf{k}_*^l(\hat{\mathbf{K}}^l)^{-1} \otimes \mathbf{W}\right)\begin{pmatrix}\mathrm{vec}(\mathbf{Y}^l)\\ \mathrm{vec}(\bar{\mathbf{Y}}^l)\end{pmatrix} + \left(\mathbf{k}_*^r(\mathbf{K}^r)^{-1} \otimes \mathbf{I}\right)\left(\mathrm{vec}(\mathbf{Y}^h) - \hat{\mathbf{W}}\begin{pmatrix}\mathrm{vec}(\mathbf{Y}^l)\\ \mathrm{vec}(\bar{\mathbf{Y}}^l)\end{pmatrix}\right), \tag{10}$$

where $\mathbf{E}_m$ and $\mathbf{E}_n$ are the selection matrices that $\hat{\mathbf{X}}^h = \mathbf{E}_m^T[\mathbf{X}^l, \hat{\mathbf{X}}^h]$, $\mathbf{X}^h = \mathbf{E}_n^T[\mathbf{X}^l, \hat{\mathbf{X}}^h]$, $\hat{\mathbf{W}} = \mathbf{E}_n^T \otimes \mathbf{W}$ and $\hat{\mathbf{K}}^l$ is the covariance matrix that $\hat{\mathbf{K}}^l = \mathbf{K}^l([\mathbf{X}^l, \hat{\mathbf{X}}^h], [\mathbf{X}^l, \hat{\mathbf{X}}^h])$.

3.4 Autokrigeability, Complexity, and Further Acceleration

For subset structured data, the computational complexity of GAR is decomposed into two independent TGPs for likelihood and predictive posterior. Thanks to the tensor algebra (mainly $(\mathbf{K} \otimes \mathbf{S})^{-1} = \mathbf{K}^{-1} \otimes \mathbf{S}^{-1}$), the complexity of the i-fidelity kernel matrix inversion is reduced to $\mathcal{O}(\sum_{m=1}^{M}(d_m^i)^3 + (N^i)^3)$ instead of $\mathcal{O}((N^i d^i)^3)$. For the non-subset case, the computational complexity in Eq. (8) is unfortunately $\mathcal{O}((N_m^i d^i)^3)$ where N_m is the number of the imaginary low-fidelity points. Nevertheless, due to the tensor structure, we can still use conjugate gradient [26] to solve the linear system efficiently.

Notice that the mean prediction $\bar{\mathbf{Z}}_*^h$ in Eq. (A.60) and Eq. (9) does not depend on any output covariance matrixes $\{\mathbf{S}_m^h, \mathbf{S}_m^l\}_{m=1}^M$, which reassemble the autokrigeability (no knowledge transfer in noiseless cases for mean predictions) [14, 9] based on the GAR framework. For applications where the predictive variation is not of interest, we can introduce a conditional independent output-correlation, i.e., $\mathbf{S}_m^h = \mathbf{I}, \mathbf{S}_m^l = \mathbf{I}$ and orthogonal weight matrixes, i.e., $\mathbf{W}_m^T\mathbf{W}_m = \mathbf{I}$, to reduce the computationally complexity further down to $\mathcal{O}((N^i)^3)$ (see Appendix for detailed proof). We call this CIGAR as an abbreviation for conditional independent GAR. In practice, CIGAR is slightly worse than GAR due to the difficulty of ensuring $\mathbf{W}_m^T\mathbf{W}_m = \mathbf{I}$.

4 Related Work

GP for high-dimensional outputs is an important model in many applications such as spatial data modeling and uncertainty quantification. For an excellent review, the readers are referred to [27]. Linear model of coregionalization (LMC) [28, 29] might be the most general framework for high-dimensional GP developed in the geostatistic community. LMC assumes that the full covariance matrix as a sum of constant matrixes timing input-dependent kernels. To reduce model complexity, semiparametric latent factor models (SLFM) [30] simplify LMC by assuming that the matrixes are rank-1 matrixes. [31] further simplifies SLFM using singular value decomposition (SVD) on the output collection to fix the bases for the rank-1 matrixes. To overcome the linear assumptions of LMC, the (implicit) bases can be constructed in a nonlinear fashion using manifold learning, e.g., KPCA [32] and IsoMap [33] or process convolution [34–36]. Other approaches include multi-task GP, which considers the outputs as dependent tasks [37–39] in a framework similar to LMC and GP regression network (GPRN) [40, 41], which proposes products of GPs to model nonlinear outputs, leading to nontractable models. Despite their success, the complicity of the above approaches are at best $\mathcal{O}(N^3 + D^3)$ whereas some are $\mathcal{O}(N^3D^3)$, which cannot scale well to high-dimensional outputs for scientific data where D can be, says, 1 million. This problem can be well resolved by introducing tensor algebra [42] to form HOGP [18] or scalable model inference, e.g., in GPRN [43].

Multi-fidelity has become a promising approach to further reduce the data demands in building a surrogate model [13] and Bayesian optimization. The seminal autoregressive (AR) model of Kennedy [3] introduce a linear transformation to univariate high-fidelity outputs. This model was enhanced by Le Gratiet [15] by adopting a deterministic parametric form of linear transformation for the efficient training scheme as introduced previously. However, it is unclear how AR can deal with high-dimensional outputs. To overcome the linearity of AR, Perdikaris et al. [16] proposes nonlinear AR (NAR). It ignores the output distributions and directly uses the low-fidelity solution as an input for the high-fidelity GP model, which is essentially a concatenating GP structure known as *deep GP* [44]. To propagate uncertainty through the multi-fidelity model, Cutajar et al. [25] uses expensive approximation inference, which makes it prone to overfitting and incapable of dealing with very large dimensional problems. For multi-fidelity Bayesian optimization (MFBO), Poloczek et al. [4]

and Kandasamy et al. [45] approximate each fidelity with a GP independently; Zhang et al. [46] use convolution kernel, similar to the process convolution [34, 36] to learn the fidelity correlations; Song et al. [5] combine all fidelity data into one single GP to reduce uncertainty. However, most MFBO surrogates do not scale to high-dimensional problems because they are designed for one target (or at most a couple).

To deal with large dimensional outputs, e.g., spatial-temporal fields, Xing et al. [9] extend AR by assuming a simple additive structure and replacing the simple GPs with scalable multi-output GPs at the cost of losing the power for capturing the output correlations, leading to inferior performance and inaccurate uncertainty estimates; Xing et al. [12] propose Deep coregionalization to extend NAR by learning the latent process [30, 29] extracted from embedding the high-dimensional outputs onto a residual latent space using a proposed residual PCA; Wang et al. [8] further introduce bases propagation along with latent process propagation in a deep GP to increase model flexibility at the cost of significant growth in the number of model parameters and a few simplifications in the approximated inference. Parussini et al. [6] generalize NAR to high-dimensional problems. However, these methods lack a systematic way for joint model training, leading to instability and poor fitting for small datasets. Wu et al. [47] extend GP using neural process to model high-dimensional and non-subset problem effectively. In scientific computing, multi-fidelity fusion has been implemented using stochastic collocation (SC) method [24] for high-dimensional problems, which provides closed-form solutions and efficient design of experiments for the multi-fidelity problem. Xing et al. [7] showed that SC is essentially a special case of AR and proposed active learning to select the best subset for the high-fidelity experiments.

To take the advances of deep learning neural network (NN) and being compatible with arbitrary multi-fidelity data (i.e., non-subset structure), Perrone et al. [22] propose an NN-based multi-task method that can naturally extend to MFBO. Li et al. [23] further extend it as a Bayesian neural network (BNN) to MFBO. Meng and Karniadakis [48] add a physics regularization layer, which requires an explicit form of the problem PDEs, to improve prediction accuracy. To scale for high-dimensional problems with arbitrary dimensions in each fidelity, Li et al. [13] propose a Bayesian network approach to multi-fidelity fusion with active learning techniques for efficiency improvement.

Except for multi-fidelity fusion, AR can be used for model multi-variate problem [49, 50], where GAR can also find its applications. GAR is a general framework for GP-based multi-fidelity fusion of high-dimensional outputs. Specifically, AR is a special case when setting $\mathbf{W} = \rho\mathbf{I}$ and using a separable kernel; ResGP is a special case of GAR by setting $\mathbf{W} = \mathbf{I}$ and $\mathbf{S} = \mathbf{I}$; NAR is a special case of integrating out $\mathbf{W}$ with a normal prior and using a separable kernel; DC is a special case of GAR if it only uses one latent process, integrating out $\mathbf{W}$ as in NAR with a separable kernel; MF-BNN is a finite case of GAR if only one hidden layer is used. See Appendix C for the comparison between SOTA methods.

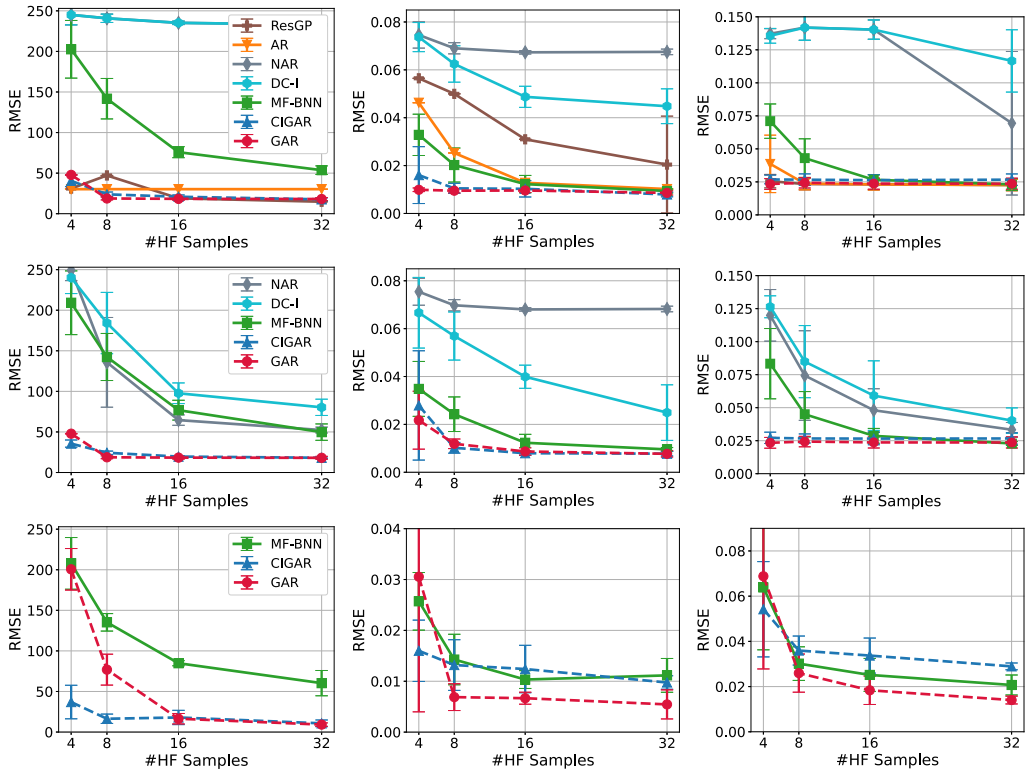
5 Experimental Results

To access GAR and CIGAR, we compare them with the SOTA multi-fidelity fusion methods for high-dimensional outputs. Particularly. we compare to the following methods: (1) AR [3], (2) NAR [16], (3) ResGP [9], (4) DC[1] [12], and, (5) MF-BNN[2] [13]. All GPs use an RBF kernel for a fair comparison. Because the ARD kernel is separable, the AR and NAR are accelerated using the Kronecker product structure as in GAR for a feasible computation. The original DC with residual PCA cannot deal with unaligned outputs, but it can do so by using an independent PCA, which we called DC-I. Both DCs preserve 0.99% energy for dimension reductions. MF-BNN is conducted using its default setting. GAR, CIGAR, AR, NAR, and ResGP are implemented using Pytorch[3]. All experiments are run on a workstation with an AMD 5950x CPU and 32 GB RAM.

5.1 Multi-Fidelity Fusion for Canonical PDEs

We first access GAR in canonical PDE simulation benchmarks, which produces high-dimensional spatial/spatial-temporal fields as model outputs. Specifically, we test on Burgers' , Poisson's and Heat equations as in [12, 51-53]. The high fidelity results are obtained by solving these equations using finite difference on a 32×32 mesh, whereas low fidelity by a 8×8 coarse mesh. The solutions on these grid points are recorded and vectorized as outputs. Because the mesh differs, the dimensionality varies. To compare with the standard multi-fidelity method that can only deal with aligned outputs, we use interpolation to upscale the low-fidelity and record them at the high-fidelity grid nodes. The

[1] https://github.com/wayXing/DC [2] https://github.com/shib0li/DNN-MFBO [3] https://pytorch.org/



(a) Poisson's equation (b) Burger's (c) Heat

Figure 1: RMSE against increasing number of high-fidelity training samples for Poisson's, Burger's, and heat equations with aligned outputs (top row), non-aligned outputs (middle row), and non-subset data (bottom row).

corresponding inputs are PDE parameters and parameterized initial or boundary conditions. Detailed experimental setups can be found in Appendix E.1

We uniformly generate 128 samples for testing and 32 for training. We increase the high-fidelity training samples to the number of low-fidelity training samples of 32. The comparisons are conducted five times with shuffled samples. The statistical results (mean and std) of the RMSE are reported in Fig. 1. GAR and CIGAR outperform competitors with a significant margin, up to 6x reduction in RMSE and also reaching the optimal with maximum 8 high-fidelity samples, indicating a successful transferring from low- to high-fidelity. CIGAR is slightly worse than GAR possibility due to lack of hard constraint on the orthogonality of its weight matrixes during implementation. As we have discovered in the literature, AR consistently performs well. With a flexible linear transformation, GAR outperforms the AR while inheriting its robustness, leading to the best performance. For the unaligned output, MF-BNN showed slightly worse performance than in the aligned cases, highlighting the challenges of the unaligned outputs. In contrast, GAR and CIGAR show almost identical performance for both cases. Nevertheless, MF-BNN also shows good performance compared to the rest of the other methods, which is consistent with the finding in [13]. It is interesting to see that for the non-subset data, the capable methods show better performances than in the subset cases. GAR and CIGAR still outperforms the competitors with a clear margin.

To approximately assess the performance under an active learning process. We instead generate training samples in a Sobol sequence [54]. The results are shown in Appendix E.2 where GAR and CIGAR also outperform the other methods by a large margin.

5.2 Multi-Fidelity Fusion for Real-World Applications

Optimal topology structure is the optimized layout of materials, e.g., alloy and concrete, given some design specifications, e.g., external force and angle. This topology optimization is a key technique in

mechanical designs, but it is also known for its high computational cost, which renders the need for multi-fidelity fusion. We consider the topology optimization of a cantilever beam with the location of point load, the angle of point load, and the filter radius [55] as system inputs. The low-fidelity use a 16×16 regular mesh for the finite element solver, whereas the high-fidelity 64×64. Please see Appendix E.3 for detailed setup.

As in the previous experiment, the RMSE statistics against an increasing number of high-fidelity training samples are shown in Fig. 2. It is clear that GAR outperforms other competitors with a large margin consistently. CIGAR can be as good as GAR when the number of training samples is large.

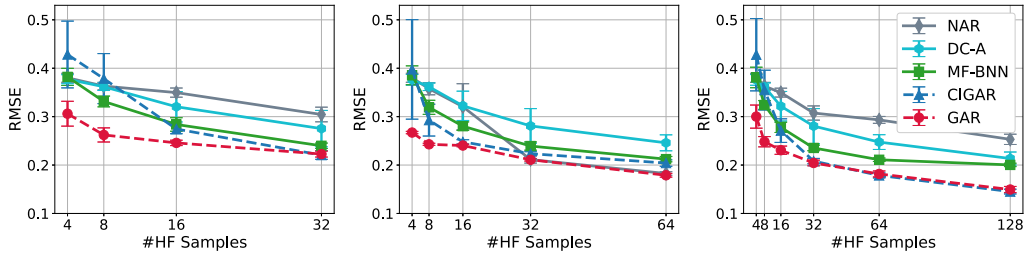
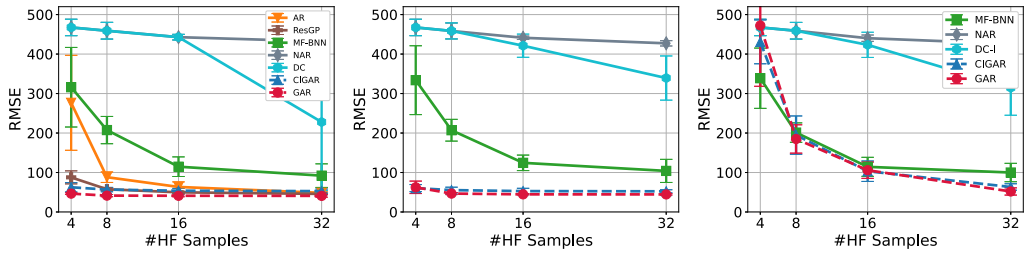


Figure 2: RMSE with low-fidelity training sample number fixed to {32,64,128}.

Steady-state 3D solid oxide fuel cell which solves complex coupled PDEs including Ohm's law, Navier-Stokes equations, Brinkman equation, and Maxwell-Stefan diffusion and convection simultaneously, is a key model for modern fuel cell optimization. The model was solved using finite element in COMSOL. The inputs were taken to be the electrode porosities, cell voltage, temperature, and pressure in the channels. The low-fidelity experiment is conducted using 3164 elements and relative tolerance of 0.1, whereas the high-fidelity uses 37064 elements and relative tolerance of 0.001. The outputs are the coupled fields of electrolyte current density (ECD) and ionic potential (IP) in the $x - z$ plane located at the center of the channel.

The RMSE statistics are shown in Fig. 3a, which, again, highlights the superiority of the proposed method with only four high-fidelity data points. To further assess the model capacity for non-structured outputs, we keep only the ECD (Fig. 3b) and IP (Fig. 3c) in the low-fidelity training data to rise the challenges of predicting high fidelity ECD+IP fields. We can see that removing some low-fidelity information indeed increases the difficulties, especially when removing ECD, where MF-BNN outperforms GAR and CIGAR with a small number of training data. As soon as the training number increases, GAR and CIGAR become superior again.

We have conducted experiments of relative low-dimensional outputs, i.e., **Plasmonic nanoparticle arrays**. GAR and CIGAR consistently outperform other methods by a large margin (Appendix E.5).



(a) ECD+IP (b) ECD (c) IP

Figure 3: RMSE for SOFC with low-fidelity training sample number fixed to 32.

6 Conclusion

We propose GAR, the first AR generalization to arbitrary outputs and non-subset multi-fidelity data with a closed-form solution,and CIGAR, an efficient implementation by revealing the autokrigeability in the AR. Limitation of this work is scalability w.r.t to samples, lack of active learning [13] and the applications to the broader problems of time series and transfer learning[49, 50]

References

[1] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando Freitas. Taking the Human Out of the Loop: A Review of Bayesian Optimization. 104(1):148–175. ISSN 0018-9219, 1558-2256. doi: 10.1109/JPROC.2015.2494218. URL https://ieeexplore.ieee.org/document/7352306/.

[2] Apostolos F. Psaros, Xuhui Meng, Zongren Zou, Ling Guo, and George Em Karniadakis. Uncertainty Quantification in Scientific Machine Learning: Methods, Metrics, and Comparisons.

[3] M. Kennedy. Predicting the output from a complex computer code when fast approximations are available. 87(1):1–13. ISSN 0006-3444, 1464-3510. doi: 10.1093/biomet/87.1.1.

[4] Matthias Poloczek, Jialei Wang, and Peter Frazier. Multi-information source optimization. Advances in neural information processing systems, 30, 2017.

[5] Jialin Song, Yuxin Chen, and Yisong Yue. A General Framework for Multi-fidelity Bayesian Optimization with Gaussian Processes. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 3158–3167. URL http://proceedings.mlr.press/v89/song19b.html.

[6] L. Parussini, D. Venturi, P. Perdikaris, and G.E. Karniadakis. Multi-fidelity Gaussian process regression for prediction of random fields. 336(C):36–50. ISSN 0021-9991. doi: 10.1016/j.jcp.2017.01.047.

[7] W. Xing, M. Razi, R. M. Kirby, K. Sun, and A. A. Shah. Greedy nonlinear autoregression for multifidelity computer models at different scales. 1:100012, . ISSN 2666-5468. doi: 10.1016/j.egyai.2020.100012. URL https://www.sciencedirect.com/science/article/pii/S2666546820300124.

[8] Zheng Wang, Wei Xing, Robert Kirby, and Shandian Zhe. Multi-Fidelity High-Order Gaussian Processes for Physical Simulation. In International Conference on Artificial Intelligence and Statistics, pages 847–855. PMLR. URL http://proceedings.mlr.press/v130/wang21c.html.

[9] W. W. Xing, A. A. Shah, P. Wang, S. Zhe, Q. Fu, and R. M. Kirby. Residual gaussian process: A tractable nonparametric bayesian emulator for multi-fidelity simulations. 97:36–56, . ISSN 0307-904X. doi: 10.1016/j.apm.2021.03.041. URL https://www.sciencedirect.com/science/article/pii/S0307904X21001724.

[10] M. Giselle Fernández-Godino, Chanyoung Park, Nam-Ho Kim, and Raphael T. Haftka. Review of multi-fidelity models.

[11] B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization. 60(3):550–591. ISSN 0036-1445. doi: 10.1137/16M1082469.

[12] Wei W. Xing, Robert M. Kirby, and Shandian Zhe. Deep coregionalization for the emulation of simulation-based spatial-temporal fields. 428:109984, . ISSN 0021-9991. doi: 10.1016/j.jcp.2020.109984. URL https://linkinghub.elsevier.com/retrieve/pii/S0021999120307580.

[13] Shibo Li, Robert M Kirby, and Shandian Zhe. Deep multi-fidelity active learning of high-dimensional outputs. arXiv preprint arXiv:2012.00901, 2020.

[14] Mauricio A. Alvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for Vector-Valued Functions: A Review.

[15] Loic Le Gratiet. Bayesian Analysis of Hierarchical Multifidelity Codes. 1(1):244–269. ISSN 2166-2525. doi: 10.1137/120884122.

[16] Paris Perdikaris, M Raissi, Andreas Damianou, N D. Lawrence, and George Karniadakis. Nonlinear Information Fusion Algorithms for Data-Efficient Multi-Fidelity Modelling, volume 473. Royal Society. doi: 10.1098/rspa.2016.0751.

[17] Carl Edward Rasmussen and Christopher K I Williams. Gaussian Processes for Machine Learning. page 266.

[18] Shandian Zhe, Wei Xing, and Robert M. Kirby. Scalable High-Order Gaussian Process Regression. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 2611–2620. PMLR. URL http://proceedings.mlr.press/v89/zhe19a.html.

[19] Tamara Gibson Kolda. Multilinear operators for higher-order decompositions. URL http://www.osti.gov/servlets/purl/923081-u0xXJa/.

[20] Zenglin Xu, Feng Yan, Yuan, and Qi. Infinite Tucker Decomposition: Nonparametric Bayesian Models for Multiway Data Analysis.

[21] Andrew Gordon Wilson and Hannes Nickisch. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, pages 1775–1784. JMLR.org.

[22] Valerio Perrone, Rodolphe Jenatton, Matthias W Seeger, and Cédric Archambeau. Scalable hyperparameter transfer learning. Advances in neural information processing systems, 31, 2018.

[23] Shibo Li, Wei Xing, Robert Kirby, and Shandian Zhe. Multi-Fidelity Bayesian Optimization via Deep Neural Networks. 33:8521–8531, . URL https://papers.nips.cc/paper/2020/hash/60e1deb043af37db5ea4ce9ae8d2c9ea-Abstract.html.

[24] Akil Narayan, Claude Gittelson, and Dongbin Xiu. A Stochastic Collocation Algorithm with Multifidelity Models. 36(2):A495–A521. ISSN 1064-8275. doi: 10.1137/130929461.

[25] Kurt Cutajar, Mark Pullin, Andreas Damianou, Neil Lawrence, and Javier González. Deep Gaussian Processes for Multi-fidelity Modeling.

[26] Andrew Wilson and Ryan Adams. Gaussian process kernels for pattern discovery and extrapolation. In International conference on machine learning, pages 1067–1075. PMLR, 2013.

[27] Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for Vector-Valued Functions: A Review. 4(3):195–266. ISSN 1935-8237, 1935-8245. doi: 10.1561/2200000036. URL http://www.nowpublishers.com/article/Details/MAL-036.

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

[29] Pierre Goovaerts et al. Geostatistics for natural resources evaluation. Oxford University Press on Demand, 1997.

[30] Yee Whye Teh, Matthias Seeger, and Michael I Jordan. Semiparametric latent factor models. In International Workshop on Artificial Intelligence and Statistics, pages 333–340. PMLR, 2005.

[31] Dave Higdon, James Gattiker, Brian Williams, and Maria Rightley. Computer Model Calibration Using High-Dimensional Output. 103(482):570–583. ISSN 0162-1459, 1537-274X. doi: 10.1198/016214507000000888.

[32] W.W. Xing, V. Triantafyllidis, A.A. Shah, P.B. Nair, and N. Zabaras. Manifold learning for the emulation of spatial fields from computational models. 326:666–690, . ISSN 0021-9991. doi: 10.1016/j.jcp.2016.07.040. URL https://linkinghub.elsevier.com/retrieve/pii/S0021999116303722.

[33] Wei Xing, Akeel A. Shah, and Prasanth B. Nair. Reduced dimensional Gaussian process emulators of parametrized partial differential equations based on Isomap. 471(2174):20140697, . ISSN 1364-5021, 1471-2946. doi: 10.1098/rspa.2014.0697.

[34] Mauricio A Álvarez, Wil Ward, and Cristian Guarnizo. Non-linear process convolutions for multi-output gaussian processes. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 1969–1977. PMLR, 2019.

[35] Phillip Boyle and Marcus Frean. Dependent gaussian processes. Advances in neural information processing systems, 17, 2004.

[36] Dave Higdon. Space and Space-Time Modeling using Process Convolutions. In Clive W. Anderson, Vic Barnett, Philip C. Chatwin, and Abdel H. El-Shaarawi, editors, Quantitative Methods for Current Environmental Issues, pages 37–56. Springer London. ISBN 978-1-4471-1171-9 978-1-4471-0657-9. doi: 10.1007/978-1-4471-0657-9_2.

[37] Edwin V Bonilla, Felix V Agakov, and Christopher KI Williams. Kernel multi-task learning using task-specific features. In Artificial Intelligence and Statistics, pages 43–50. PMLR, 2007.

[38] Barbara Rakitsch, Christoph Lippert, Karsten Borgwardt, and Oliver Stegle. It is all in the noise: Efficient multi-task gaussian process inference with structured residuals. Advances in neural information processing systems, 26, 2013.

[39] Ping Li and Songcan Chen. Hierarchical Gaussian Processes model for multi-task learning. 74:134–144. ISSN 0031-3203. doi: 10.1016/j.patcog.2017.09.021. URL https://linkinghub.elsevier.com/retrieve/pii/S0031320317303746.

[40] Andrew Gordon Wilson, David A. Knowles, and Zoubin Ghahramani. Gaussian process regression networks. In Proceedings of the 29th International Coference on International Conference on Machine Learning, ICML'12, page 1139–1146, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

[41] Trung Nguyen and Edwin Bonilla. Efficient variational inference for gaussian process regression networks. In Artificial Intelligence and Statistics, pages 472–480. PMLR, 2013.

[42] Tamara G. Kolda and Brett W. Bader. Tensor Decompositions and Applications. 51(3):455–500. ISSN 0036-1445, 1095-7200. doi: 10.1137/07070111X.

[43] Shibo Li, Wei Xing, Robert M. Kirby, and Shandian Zhe. Scalable Gaussian Process Regression Networks. volume 3, pages 2456–2462, . doi: 10.24963/ijcai.2020/340. URL https://www.ijcai.org/proceedings/2020/340.

[44] Andreas Damianou. Deep Gaussian processes and variational propagation of uncertainty.

[45] Kirthevasan Kandasamy, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process bandit optimisation with multi-fidelity evaluations. Advances in neural information processing systems, 29, 2016.

[46] Yehong Zhang, Trong Nghia Hoang, Bryan Kian Hsiang Low, and Mohan Kankanhalli. Information-based multi-fidelity bayesian optimization. In NIPS Workshop on Bayesian Optimization, 2017.

[47] Dongxia Wu, Matteo Chinazzi, Alessandro Vespignani, Yi-An Ma, and Rose Yu. Multi-fidelity hierarchical neural processes. arXiv preprint arXiv:2206.04872, 2022.

[48] Xuhui Meng and George Em Karniadakis. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems. 401:109020. ISSN 0021-9991. doi: 10.1016/j.jcp.2019.109020. URL http://www.sciencedirect.com/science/article/pii/S0021999119307260.

[49] James Requeima, William Tebbutt, Wessel Bruinsma, and Richard E Turner. The gaussian process autoregressive regression model (gpar). In The 22nd International Conference on Artificial Intelligence and Statistics, pages 1860–1869. PMLR, 2019.

[50] Rui Xia, Wessel Bruinsma, William Tebbutt, and Richard E Turner. The gaussian process latent autoregressive model. In Third Symposium on Advances in Approximate Bayesian Inference, 2020.

[51] Rui Tuo, C. F. Jeff Wu, and Dan Yu. Surrogate Modeling of Computer Experiments With Different Mesh Densities. 56(3):372–380. ISSN 0040-1706, 1537-2723. doi: 10.1080/00401706.2013.842935.

[52] Mehmet Onder Efe and Hitay Ozbay. Proper orthogonal decomposition for reduced order modeling: 2d heat flow. In Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003., volume 2, pages 1273–1277. IEEE.

[53] Maziar Raissi and George Em Karniadakis. Machine Learning of Linear Differential Equations using Gaussian Processes. 348:683–693. ISSN 0021-9991. doi: 10/gbzfgr.

[54] I.M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. 7(4):86–112. ISSN 0041-5553. doi: 10.1016/0041-5553(67)90144-9. URL https://linkinghub.elsevier.com/retrieve/pii/0041555367901449.

[55] Tyler E. Bruns and Daniel A. Tortorelli. Topology optimization of non-linear elastic structures and compliant mechanisms. Computer Methods in Applied Mechanics and Engineering, 190(26):3443 – 3459, 2001. ISSN 0045-7825. doi: https://doi.org/10.1016/S0045-7825(00)00278-4. URL http://www.sciencedirect.com/science/article/pii/S0045782500002784.

[56] TJ Chung. Computational fluid dynamics. Cambridge university press.

[57] N Sugimoto. Burgers equation with a fractional derivative; hereditary effects on nonlinear acoustic waves. 225:631–653.

[58] Kai Nagel. Particle hopping models and traffic flow theory. 53(5):4655.

[59] S Kutluay, AR Bahadir, and A Özdec̄s. Numerical solution of one-dimensional burgers equation: explicit and exact-explicit finite difference methods. 103(2):251–261.

[60] A. A. Shah, W. W. Xing, and V. Triantafyllidis. Reduced-order modelling of parameter-dependent, linear and nonlinear dynamic partial differential equation models. 473(2200):20160809. ISSN 1364-5021, 1471-2946. doi: 10.1098/rspa.2016.0809.

[61] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations.

[62] Steven C Chapra, Raymond P Canale, et al. Numerical methods for engineers. Boston: McGraw-Hill Higher Education,.

[63] S Persides. The laplace and poisson equations in schwarzschild's space-time. 43(3):571–578.

[64] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial Neural Networks for Solving Ordinary and Partial Differential Equations. 9(5):987–1000, Sept./1998. ISSN 1045-9227. doi: 10.1109/72.712178.

[65] Frank Spitzer. Electrostatic capacity, heat flow, and brownian motion. 3(2):110–121.

[66] Krzysztof Burdzy, Zhen-Qing Chen, John Sylvester, et al. The heat equation and reflected brownian motion in time-dependent domains. 32(1B):775–804.

[67] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. 81(3):637–654.

[68] Wei Xing, Shireen Y. Elhabian, Vahid Keshavarzzadeh, and Robert M. Kirby. Shared-Gaussian Process: Learning Interpretable Shared Hidden Structure Across Data Spaces for Design Space Analysis and Exploration. 142(8), . ISSN 1050-0472, 1528-9001. doi: 10.1115/1.4046074.

[69] Erik Andreassen, Anders Clausen, Mattias Schevenels, Boyan S. Lazarov, and Ole Sigmund. Efficient topology optimization in matlab using 88 lines of code. Structural and Multidisciplinary Optimization, 43 (1):1–16, Jan 2011. ISSN 1615-1488. doi: 10.1007/s00158-010-0594-7. URL https://doi.org/10.1007/s00158-010-0594-7.

[70] Martin Philip Bendsoe and Ole Sigmund. Topology optimization: Theory, methods and applications. Springer, 2004.

[71] Charles-Antoine Guérin, Pierre Mallet, and Anne Sentenac. Effective-medium theory for finite-size aggregates. JOSA A, 23(2):349–358, 2006.

[72] Mani Razi, Ren Wang, Yanyan He, Robert M. Kirby, and Luca Dal Negro. Optimization of Large-Scale Vogel Spiral Arrays of Plasmonic Nanoparticles. 14(1):253–261. ISSN 1557-1955, 1557-1963. doi: 10.1007/s11468-018-0799-y.

[73] Aristi Christofi, Felipe A Pinheiro, and Luca Dal Negro. Probing scattering resonances of vogel's spirals with the green's matrix spectral method. Optics letters, 41(9):1933–1936, 2016.

[74] Dongxia Wu, Liyao Gao, Xinyue Xiong, Matteo Chinazzi, Alessandro Vespignani, Yi-An Ma, and Rose Yu. Quantifying uncertainty in deep spatiotemporal forecasting. arXiv preprint arXiv:2105.11982, 2021.

References follow the acknowledgments. Use unnumbered first-level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to `small` (9 point) when listing the references. Note that the Reference section does not count towards the page limit.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See contributions, abstract, and introduction

(b) Did you describe the limitations of your work? [Yes] See the conclusion and complexity analysis section

(c) Did you discuss any potential negative societal impacts of your work? [N/A] We do not see an obvisou negative societal impacts as it is fundamental and quite theoretical.

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [Yes]

(b) Did you include complete proofs of all theoretical results? [Yes] Please see Appendix

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Please see supplementary materials

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please see the Appendix

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Please see the experimental section

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please see the experimental section

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] Please see the experimental section

(b) Did you mention the license of the assets? [Yes]

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] see experiment senction and the Appendix

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We do not use such data

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]