SIAM/ASA J. UNCERTAINTY QUANTIFICATION
Vol. 12, No. 2, pp. 473–502

Conglomerate Multi-fidelity Gaussian Process Modeling, with Application to Heavy-Ion Collisions[*]

Yi Ji[†‖], Henry Shaowu Yuchi[‡‖], Derek Soeder[§], J.-F. Paquet[§¶], Steffen A. Bass[§], V. Roshan Joseph[‡], C. F. Jeff Wu[‡], and Simon Mak[#]

Abstract. In an era where scientific experimentation is often costly, multi-fidelity emulation provides a powerful tool for predictive scientific computing. While there has been notable work on multi-fidelity modeling, existing models do not incorporate an important "conglomerate" property of multi-fidelity simulators, where the accuracies of different simulator components are controlled by different fidelity parameters. Such conglomerate simulators are widely encountered in complex nuclear physics and astrophysics applications. We thus propose a new CONglomerate multi-FIdelity Gaussian process (CONFIG) model, which embeds this conglomerate structure within a novel non-stationary covariance function. We show that the proposed CONFIG model can capture prior knowledge on the numerical convergence of conglomerate simulators, which allows for cost-efficient emulation of multi-fidelity systems. We demonstrate the improved predictive performance of CONFIG over state-of-the-art models in a suite of numerical experiments and two applications, the first for emulation of cantilever beam deflection and the second for emulating the evolution of the quark-gluon plasma, which was theorized to have filled the universe shortly after the Big Bang.

Key words. Bayesian nonparametrics, computer experiments, multi-fidelity modeling, surrogate modeling, quark-gluon plasma

MSC codes. 62G08, 60G15

DOI. 10.1137/22M1525004

1. Introduction.

Computer experimentation is widely used for modeling complex scientific and engineering systems, particularly when physical experiments are costly, unethical, or impossible to perform. This shift from physical to computer experimentation has found success in a wide range of physical science applications, including rocket design [39], solar

[*] Received by the editors September 27, 2022; accepted for publication (in revised form) February 14, 2024; published electronically May 30, 2024.
https://doi.org/10.1137/22M1525004
Funding: The work of the first and eighth authors was supported by DOE grant DE-SC0024477 and NSF grants DMS-2210729 and DMS-2316012. The work of the first, third, and eighth authors was supported by NSF grant OAC-1550225. The work of the third, fourth, and fifth authors was supported by DOE grant DE-FG02-05ER41367.

[†] Department of Statistical Science, Duke University, Durham, NC 27708-0251 USA (yi.ji@duke.edu).

[‡] H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30308 USA (shaowu.yuchi@gatech.edu, roshan@gatech.edu, jeff.wu@isye.gatech.edu).

[§] Department of Physics, Duke University, Durham, NC 27708 USA (derek.soeder@duke.edu, jeanfrancois.paquet@duke.edu, bass@duke.edu).

[¶] Department of Physics and Astronomy & Department of Mathematics, Vanderbilt University, Nashville, TN 37240 USA.

[‖] Joint first authors.

[#] Corresponding author. Department of Statistical Science, Duke University, Durham, NC 27708-0251 USA (sm769@duke.edu).

irradiance modeling [62], and 3D printing [7]. However, as systems become more complex and realistic, such computer experiments also become more expensive, thus placing a heavy computational burden on design exploration and optimization. Statistical *emulators* [57] have shown great promise in tackling this limitation. The idea is simple but effective: computer experiments are first performed at carefully chosen design points and then used as training data to fit an *emulator* model to efficiently predict and quantify uncertainty on the expensive virtual experiment.

In recent years, however, with the increasing sophistication of modern scientific problems, an emerging challenge for emulators is the simulation of high-fidelity training data, which can be prohibitively expensive. One way to address this is via *multi-fidelity emulation,* which makes use of training simulation data of multiple *fidelities* (or accuracies) for model fitting. Such multi-fidelity data can often be generated by varying different *fidelity parameters,* which control the precision of the numerical experiment. There are a wide variety of fidelity parameters, ranging from mesh sizes for finite element analysis [41, 47] to time-steps for dynamical system simulation [67]. The goal is to leverage information from lower-fidelity (but cheaper) simulations to enhance predictions for the high-fidelity (but expensive) model, thus allowing for improved emulation and uncertainty quantification (for the highest-fidelity code) at lower computational costs.

There has been much recent work on multi-fidelity emulation, particularly for Gaussian process (GP) modeling. This includes the seminal work [33], which presented a first-order autoregressive model for integrating information over a hierarchy of simulation models from lowest to highest fidelity. This Kennedy–O'Hagan model has then been extended in various works, including a Bayesian hierarchical implementation in [54], the multi-fidelity optimization in [16], and the nonlinear fusion model in [48]. In [66], the authors proposed a multi-fidelity emulator for finite element analysis (FEA), which utilizes the discretization mesh size as the single fidelity parameter. This emulator models the bias induced by discretization mesh elements via a GP and is related to the state-of-the-art grid convergence index approach typically employed in FEA [1, 56]. Such multi-fidelity models have been widely applied in engineering design and scientific computing; see, e.g., [29, 34, 58]. Experimental design for such emulators have been explored [70], including a sequential design strategy in [23]. Similar ideas have also been applied for broader applications in data fusion [21], Bayesian optimization [43, 52], and transfer learning [65].

The above methods, however, have limitations when applied to our motivating nuclear physics application. Here, we are studying the quark-gluon plasma (QGP), a deconfined phase of nuclear matter consisting of elementary quarks and gluons. The QGP was theorized to have filled the universe shortly after the Big Bang, and the study of this plasma sheds light on the properties of this unique phase of matter. This plasma can be simulated at a small scale by virtually colliding heavy ions together at near-light speeds in particle colliders. Simulating such collisions requires a "conglomerate" system of complex dynamical models to faithfully capture the detailed evolution of the plasma. Consider, in particular, the three-stage simulation framework in [15] (see also [12, 18, 24]), which models the initial energy disposition of the heavy ions, the hydrodynamic evolution of the plasma after the collision, and the subsequent conversion of nuclear fluid into particles. Figure 1 visualizes this conglomerate (specifically, multi-stage) procedure. At each stage, the simulation of the component physics can involve

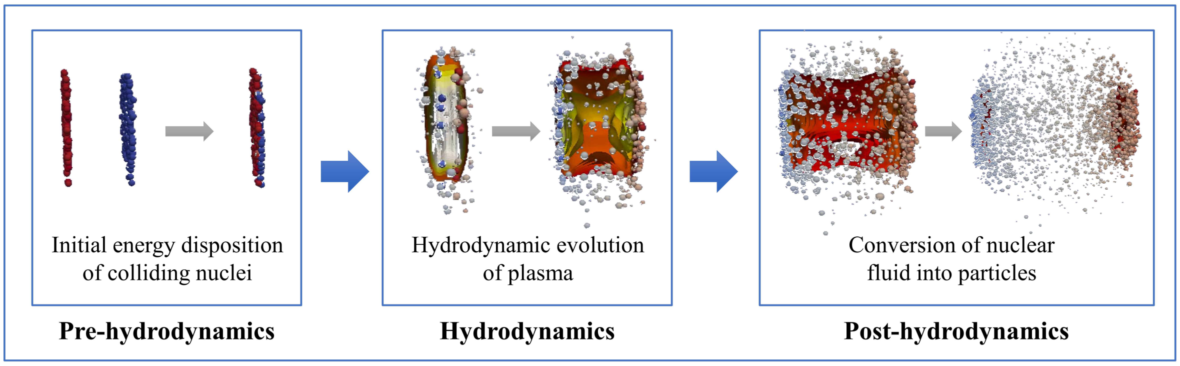

Initial energy disposition of colliding nuclei
Pre-hydrodynamics
Hydrodynamic evolution of plasma
Hydrodynamics
Conversion of nuclear fluid into particles
Post-hydrodynamics

Figure 1. *Three-stage simulation of the quark-gluon plasma.*

multiple and *different* fidelity parameters, controlling, e.g., the size of the hydrodynamics spatial mesh or the time-scale for dynamic evolution.

This *conglomerate multi-fidelity* framework, where the simulator comprises multiple submodels for simulating different physics of a complex phenomenon, poses several challenges for existing multi-fidelity emulators. First, since there are multiple fidelity parameters to set for each simulation stage, the resulting simulation runs typically cannot be ranked from lowest to highest fidelity, which is required for a direct application of Kennedy–O'Hagan-type models. For example, to gauge the effects of three fidelity parameters, the physicist may choose to run the simulator in three different ways, each with higher fidelity at one stage and lower fidelity at the remaining stages. A priori, it is unclear if these three simulation approaches can be ranked from lowest to highest fidelity. Second, unlike the multi-fidelity emulator in [66] (which allows only one fidelity parameter), there are *multiple* fidelity parameters that should be accounted for when training emulators with conglomerate simulations. Neglecting this conglomerate structure for emulation can result in significantly poorer predictive performance, as we show later. A broader emulation model is thus needed to tackle the challenges presented by conglomerate multi-fidelity simulators, which are widely encountered in nuclear physics [28] and astrophysics [26].

We propose in this work a new GP emulator that addresses these challenges. The proposed CONglomerate multi-FIdelity Gaussian process (CONFIG) model makes use of a novel *non-stationary* covariance function, which captures prior information on the numerical convergence of conglomerate simulators. Our emulator is applicable for a variety of complex multi-physics simulators, where each physics (with its corresponding fidelity parameters) is jointly simulated via a conglomerate framework. By embedding this underlying conglomerate structure within its kernel specification, the CONFIG model can yield improved emulation performance and uncertainty quantification over existing methods for predicting the limiting highest-fidelity simulator. This is demonstrated in a suite of numerical experiments, a beam deflection problem in finite element analysis, and an application to the motivating heavy-ion collision problem. Section 2 reviews several existing multi-fidelity emulators and outlines the motivating QGP problem. Section 3 presents the model specification for the proposed CONFIG emulator. Section 4 discusses implementation details for CONFIG, including parameter estimation and experimental design. Section 5 compares the proposed model with existing methods on a suite of numerical experiments. Finally, section 6 demonstrates the effectiveness of CONFIG for the motivating QGP application as well as a cantilever beam deflection problem. Section 7 concludes the paper.

2. Preliminaries and motivation.

In this section, we first provide an overview of conglomerate multi-fidelity simulators and their use for complex multi-physics applications. We then briefly introduce the GP model and review the Kennedy–O'Hagan model in [33] and the multi-fidelity model in [66]. Finally, we discuss the limitations of such models for our QGP application, thus motivating the proposed CONFIG model.

2.1. Conglomerate multi-fidelity simulation.

With an urgent need for reliable simulation of complex phenomena involving multiple physical mechanisms and/or components, conglomerate multi-fidelity simulations are now increasingly used in modern scientific and engineering applications, such as structural studies [13, 63], engine combustion [44], and high-energy physics [35]. Such simulators model the complex phenomenon via either multiple submodels that account for different physics (e.g., hydrodynamic evolution, nuclear particlization) or multiple components (e.g., spatial mesh, time discretization) that are required by the simulation procedure. Consequently, the simulation of the *overall* phenomenon typically involves *multiple* fidelity parameters, each controlling the simulation accuracy of individual submodels or components. This poses a key challenge for existing emulator models.

To tackle this, it is useful to first understand different types of conglomerate simulators encountered in applications. In our experience, this falls roughly into two scenarios (see Figure 2):

- **Scenario 1**: The simulator consists of multiple fidelity parameters for simulating a *single* mechanism or phenomenon. Such parameters control different means for varying simulation precision, e.g., via spatial meshing or temporal discretization. One example is the FEA of a cantilever beam deflection under stress, where three mesh fidelity parameters can be used for each dimension of the three-dimensional finite element analysis. We will investigate this application further in section 6.1.

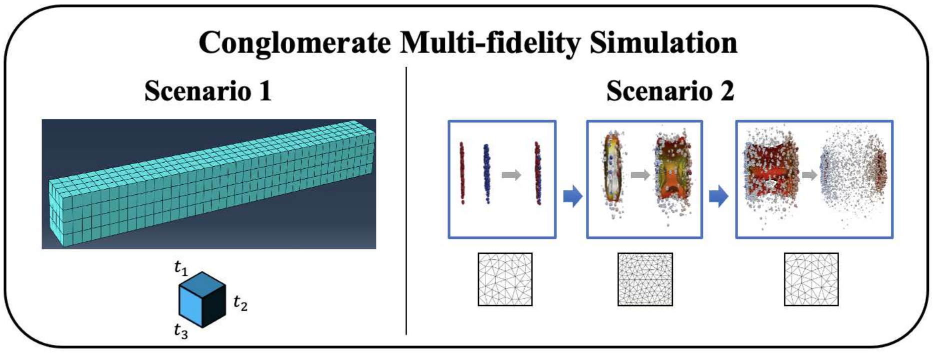


Figure 2. *Visualizing examples of Scenarios* 1 and 2 *for conglomerate multi-fidelity simulations. The left plot shows an example of Scenario* 1 *for cantilever beam deflection, where the three fidelity parameters specify the size of the finite elements for the beam. The right plot shows an example of Scenario* 2 *for heavy-ion collisions, where different fidelity parameters control simulation precision at different stages of the collision system.*

- **Scenario 2**: The simulator comprises multiple stages that are performed *sequentially* over time, where a separate phenomenon is simulated at each stage, with associated fidelity parameters. *Multiple* mechanisms are thus involved in simulating the desired phenomenon. This is the case for our motivating nuclear physics problem (Figure 1), where multiple mesh size parameters control simulation precision in each of the three consecutive stages for heavy-ion collisions. We will investigate this application further in section 6.2.

Motivated by these two scenarios, we will present later two variations of the CONFIG model that tackle each of these scenarios; more on this is presented in section 3.

2.2. Gaussian process modeling. Gaussian process (GP) modeling is a popular Bayesian non-parametric approach for supervised learning [68] with broad applications for computer experiments [57]. The specification of a GP model involves two key ingredients: the mean function and the covariance function. Let $\mathbf{x} \in [0,1]^p$ be the input parameters (sufficiently scaled) for the simulator, and let $\eta(\mathbf{x})$ be the corresponding output of the simulator. (In practice, the inputs need not be confined to a hypercube and can be defined beyond Euclidean spaces via arbitrary index sets.) A GP model places the following prior on the unknown response surface $\eta(\cdot)$:

$$\eta(\cdot) \sim \mathcal{GP}(\mu(\cdot), k(\cdot,\cdot)). \tag{2.1}$$

Here, $\mu(\cdot)$ is the mean function controlling the centrality of the stochastic process. If appropriate basis functions $\boldsymbol{f}(\boldsymbol{x})$ are known, one can model the mean function as $\mu(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x})^T\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ are the corresponding coefficients on $\boldsymbol{f}(\boldsymbol{x})$. In the absence of such information, $\mu(\cdot)$ is typically set to be a constant. The function $k(\cdot,\cdot)$ is the covariance function that controls the smoothness of its sample paths. Common choices of $k(\cdot,\cdot)$ include the squared-exponential and Matérn kernels [57].

Let $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ denote the simulated input points, and let $\mathbf{y} = [\eta(\mathbf{x}_1), \ldots, \eta(\mathbf{x}_n)]$ be the simulated outputs. Assuming that the kernel hyperparameters are fixed and known (we will discuss the estimation of such parameters later in section 4.1), the predictive distribution $\eta(\mathbf{x}^*)$ at the new input $\mathbf{x}^*$ conditional on data $\{\mathcal{D}, \mathbf{y}\}$ is given by

$$\eta(\mathbf{x}^*)|\mathcal{D}, \mathbf{y} \sim \mathcal{GP}(\hat{\mu}(\mathbf{x}^*), s^2(\mathbf{x}^*)). \tag{2.2}$$

Here, the posterior mean and variance are given by

$$\begin{aligned} \hat{\mu}(\mathbf{x}^*) &= \mu(\mathbf{x}^*) + \mathbf{k}(\mathbf{x}^*, \mathcal{D})^T \mathbf{K}(\mathcal{D})^{-1}(\mathbf{y} - \boldsymbol{\mu}(\mathcal{D})), \\ s^2(\mathbf{x}^*) &= k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*, \mathcal{D})^T \mathbf{K}(\mathcal{D})^{-1}\mathbf{k}(\mathbf{x}^*, \mathcal{D}), \end{aligned} \tag{2.3}$$

where $\mathbf{k}(\mathbf{x}^*, \mathcal{D}) = [k(\mathbf{x}^*, \mathbf{x}_1), \ldots, k(\mathbf{x}^*, \mathbf{x}_n)]$ is the vector of covariances, $\boldsymbol{\mu}(\mathcal{D}) = [\mu(\mathbf{x}_1), \ldots, \mu(\mathbf{x}_n)]$ is the vector of means, and $\mathbf{K}(\mathcal{D})$ is the covariance matrix for the training data. The models introduced later in this paper will make use of these closed-form predictive equations with different choices of covariance functions.

2.3. The Kennedy–O'Hagan model. In the seminal work of [33], the authors proposed a first-order autoregressive model for linking outputs from a hierarchy of H simulators from the lowest fidelity (simulator 1) to the highest fidelity (simulator H). Let $\eta_h(\mathbf{x})$ denote the output from simulator h at standardized input parameters $\mathbf{x} \in [0,1]^p$. The Kennedy–O'Hagan (KOH) model is specified as

$$\eta_h(\mathbf{x}) = \rho_{h-1}\eta_{h-1}(\mathbf{x}) + \delta_h(\mathbf{x}), \quad h = 2, \ldots, H. \tag{2.4}$$

Here, ρ_{h-1} is a regression scale factor, and $\delta_h(\mathbf{x})$ is a bias term that models the discrepancy between simulator $h-1$ and h. The bias term $\delta_h(\mathbf{x})$ may be modeled by a stationary GP with a squared-exponential covariance function [57]

$$\mathrm{Cov}\left[\delta_h(\mathbf{x}), \delta_h(\mathbf{x}')\right] = \sigma_h^2 \exp\left\{-\sum_{i=1}^{p} \phi_{h,i}(\mathbf{x}_i - \mathbf{x}_i')^2\right\}, \tag{2.5}$$

where $\phi_{h,i}$ is the weight parameter for the ith input parameter at the hth fidelity level. Such a model allows one to integrate information from a sequence of simulator models with varying fidelity levels to efficiently emulate the highest-fidelity simulator model.

The KOH multi-fidelity model has subsequently been extended in a variety of ways, including a Bayesian implementation in [53] and a nonlinear extension in [49]; see also [13, 17, 55]. This modeling framework is also closely related to the idea of co-kriging [60] in spatial statistics and was employed for sequential co-kriging design [36]. However, the aforementioned methods assume that the multi-fidelity training data can be *ranked* from lowest to highest fidelity. As such, this body of literature does not directly apply to the motivating problem of conglomerate multi-fidelity emulation, where simulation accuracy is controlled by *multiple* fidelity parameters, and thus there is no clear ranking of training data from lowest to highest fidelity. There are several ways to force existing models on this problem, but each has its shortcomings. One could design the data such that the training simulations are ranked (e.g., increasing all fidelity parameters simultaneously), but this would result in highly inefficient designs which fail to sufficiently explore the space of fidelity parameters. One could also arbitrarily assign a *single* "artificial" fidelity level for each simulation, which imposes a ranking on the training runs. This, however, *neglects* the rich conglomerate multi-fidelity framework (i.e., the "science") for the simulator, which can lead to significantly poorer predictive performance from the emulator, as we show later.

2.4. The Tuo–Wu–Yu model. For problems where the fidelity level is controlled by a *single* continuous fidelity parameter t (e.g., mesh size), an alternative model is proposed in [66] (we call this the TWY model) that can make use of such information. Let $\eta(\mathbf{x}, t)$ denote the deterministic code output at standardized inputs $\mathbf{x} \in [0,1]^p$ and at fidelity parameter t. Here, t is typically assumed to be between 0 and 1, i.e., $t \in [0,1]$, with a smaller t indicating a finer mesh size or, equivalently, higher mesh density. The TWY model adopts the following model for $\eta(\mathbf{x}, t)$:

$$\eta(\mathbf{x}, t) = \eta(\mathbf{x}, 0) + \delta(\mathbf{x}, t) =: \phi(\mathbf{x}) + \delta(\mathbf{x}, t). \tag{2.6}$$

Here, $\phi(\mathbf{x}) := \eta(\mathbf{x}, 0)$ denotes the "exact" simulation output at input $\mathbf{x}$ at the highest (limiting) fidelity $t = 0$, and $\delta(\mathbf{x}, t)$ denotes the discrepancy (or bias) between this exact solution and realized simulation output with mesh size t. In practical problems, the exact solution $\eta(\mathbf{x}, 0)$

is typically *not obtainable* numerically since some level of approximation (e.g., mesh or time discretization) is needed for simulating the system. The goal is to leverage simulation training data of the form $\{\eta(\mathbf{x}_i, t_i)\}_{i=1}^n$ along with an appropriate model on (2.6) to predict the exact solution $\eta(\mathbf{x}, 0)$.

Since $\phi(\mathbf{x})$ and $\delta(\mathbf{x}, t)$ are unknown a priori, these terms are modeled in [66] by two independent Gaussian processes. For $\phi(\mathbf{x})$, a standard GP prior is assigned with constant mean and a stationary correlation (e.g., squared-exponential) function. For the bias term $\delta(\mathbf{x}, t)$, a *nonstationary* zero-mean GP prior is assigned with covariance function

$$\text{Cov}[\delta(\mathbf{x_1}, t_1), \delta(\mathbf{x_2}, t_2)] = \sigma_2^2 K_{\mathbf{x}}^{\delta}(\mathbf{x}_1, \mathbf{x}_2) \min(t_1, t_2)^{\ell}, \tag{2.7}$$

where $K_{\mathbf{x}}^{\delta}(\cdot, \cdot)$ is a stationary correlation function on input parameters $\mathbf{x}$, and ℓ is a hyperparameter. One can view this as a product of two kernels, where the kernel on the fidelity parameter t is non-stationary and closely resembles that of a Brownian motion. This separable kernel structure has been utilized in [50] as well.

This choice of non-stationary kernel over the single fidelity parameter t can be reasoned from a Bayesian modeling perspective. Consider the GP model with covariance function (2.7) as a prior model on discrepancy $\delta(\mathbf{x}, t)$. Before observing the data, one can show from (2.7) that

$$\lim_{t \to 0} \delta(\mathbf{x}, t) = 0 \quad \text{for all } \mathbf{x} \in [0, 1]^p. \tag{2.8}$$

The TWY model thus assumes a priori that the discrepancy term should converge to 0 as fidelity parameter t goes to 0 or, equivalently, that the simulation output $\eta(\mathbf{x}, t)$ converges to the exact solution $\phi(\mathbf{x})$ as we increase the fidelity of the simulator. This can be seen as a way of integrating *prior* information on the numerical convergence of the simulator within the prior specification of the emulator model. One can further set the kernel parameter ℓ to capture additional information on known numerical convergence rates of the simulator; see [66] for details.

For the target conglomerate setting where *multiple* fidelity parameters are present, the TWY model needs to be further extended. A simple modification might be to first assign for each simulation run an "artificial" fidelity, e.g., the average of the multiple fidelity parameters, and then use this single aggregate fidelity level with the TWY model for multi-fidelity emulation. However, such an approach ignores the rich conglomerate structure of the simulation framework, which can lead to poor predictive performance. We show later that, by integrating directly the conglomerate multi-fidelity nature of the simulation framework (i.e., the "science") within the CONFIG model, we can achieve significantly improved predictive performance in numerical experiments and for the motivating nuclear physics application.

3. The CONFIG model.

Given these limitations, we now present the proposed CONFIG model for the efficient emulation of conglomerate multi-fidelity simulations. Our model adopts a novel non-stationary GP model which captures *prior* information on the numerical convergence behavior of *conglomerate* simulators. Below, we outline the general CONFIG model specification and then present two choices of non-stationary covariance functions which capture this desired prior information.

Let $\mathbf{x} \in [0,1]^p$ be the vector of p standardized simulation inputs for the computer code (again assumed to be deterministic), and suppose there are q fidelity parameters (denoted by $\mathbf{t} \in [0,1]^q$) that control simulation accuracy in the code. These may, e.g., consist of different mesh sizes for domain discretization and time steps at different simulation stages. As before, a smaller fidelity parameter t_r (with other fidelity parameters held constant) yields more accurate simulations at higher computational costs, with $t_r = 0$ denoting the highest (limiting) fidelity level. Let $\eta(\mathbf{x},\mathbf{t})$ denote the deterministic code output at inputs $\mathbf{x}$ and fidelity parameters $\mathbf{t}$. The CONFIG model assumes the following decomposition of $\eta(\mathbf{x},\mathbf{t})$:

$$\eta(\mathbf{x},\mathbf{t}) = \eta(\mathbf{x},\mathbf{0}) + \delta(\mathbf{x},\mathbf{t}) := \phi(\mathbf{x}) + \delta(\mathbf{x},\mathbf{t}). \tag{3.1}$$

Similar to before, $\phi(\mathbf{x}) := \eta(\mathbf{x},\mathbf{0})$ models the "exact" simulation solution at the highest (limiting) fidelity setting of $\mathbf{t} \to \mathbf{0}$, and $\delta(\mathbf{x},\mathbf{t})$ models the numerical discrepancy (or error) between the exact solution $\phi(\mathbf{x})$ and the simulated output $\eta(\mathbf{x},\mathbf{t})$. We next place independent GP priors on both terms. For $\phi(\mathbf{x})$, a standard GP is assigned with user-defined basis functions for the mean and a stationary correlation function. In our later implementation, we make use of linear basis functions along with the popular squared-exponential correlation function

$$\mathrm{Cov}[\phi(\mathbf{x}_1),\phi(\mathbf{x}_2)] = \sigma_1^2 K_{\mathbf{x}}^{\phi}(\mathbf{x}_1,\mathbf{x}_2) = \sigma_1^2 \exp\left\{-\sum_{s=1}^{p} \gamma_s (x_{1,s} - x_{2,s})^2\right\}, \tag{3.2}$$

where γ_s is the weight parameter for the sth input dimension.

For the bias term $\delta(\mathbf{x},\mathbf{t})$, we will carefully specify a new nonstationary covariance function that captures one's *prior* knowledge on the numerical convergence behavior. One desirable property of $\delta(\mathbf{x},\mathbf{t})$ is the limiting constraint

$$\lim_{\mathbf{t}\to\mathbf{0}} \delta(\mathbf{x},\mathbf{t}) = 0 \quad \text{for all } \mathbf{x} \in [0,1]^p. \tag{3.3}$$

In words, for any inputs $\mathbf{x}$, the simulation output $\eta(\mathbf{x},\mathbf{t})$ should converge to the underlying exact solution $\phi(\mathbf{x})$ when *all* fidelity parameters converge to zero, i.e., all fidelity levels are set to their highest (limiting) setting. Property (3.3) should thus be satisfied if the simulator enjoys theoretical convergence guarantees (e.g., weak convergence of PDE solutions) or is trusted to converge empirically. Another desirable property is that, for a fidelity parameter t_r and fixed levels of the remaining fidelity parameters $\mathbf{t}_{-r} \neq \mathbf{0}$, we have

$$\lim_{t_r\to 0} \mathbb{P}(\delta(\mathbf{x},\mathbf{t}) = 0) = 0 \quad \text{for all } \mathbf{x} \in [0,1]^p. \tag{3.4}$$

In words, for any inputs $\mathbf{x}$ and if any fidelity parameters $\mathbf{t}_{-r}$ are nonzero, there should be a nonnegligible (i.e., nonzero) discrepancy between the simulation output $\eta(\mathbf{x},\mathbf{t})$ and the underlying true solution $\phi(\mathbf{x})$. This is again intuitive, as the simulator should not reach the true solution when some of its fidelity parameters are not at their highest fidelities. The two limiting constraints thus describe how fidelity parameters determine the discrepancy behavior of the simulator: only when *all* fidelity parameters approach zero should the simulator converge to the true solution.

To satisfy these two properties, we place a GP prior on $\delta(\mathbf{x},\mathbf{t})$ with product covariance form

$$\mathrm{Cov}[\delta(\mathbf{x_1},\mathbf{t}_1),\delta(\mathbf{x_2},\mathbf{t}_2)] = \sigma_2^2 K_{\mathbf{x}}^{\delta}(\mathbf{x}_1,\mathbf{x}_2)K_{\mathbf{t}}^{\delta}(\mathbf{t}_1,\mathbf{t}_2); \tag{3.5}$$

i.e., the effect of input variables and fidelity parameters are assumed to be separable for δ. For the first kernel $K_{\mathbf{x}}^{\delta}(\cdot,\cdot)$, one can employ a standard stationary kernel; we make use of the squared-exponential kernel

$$K_{\mathbf{x}}^{\delta}(\mathbf{x}_1,\mathbf{x}_2) = \exp\left\{-\sum_{s=1}^{p}\alpha_s(x_{1,s}-x_{2,s})^2\right\} \tag{3.6}$$

in our later implementation. For the second kernel $K_{\mathbf{t}}(\mathbf{t}_1,\mathbf{t}_2)$, a careful *non-stationary* specification is needed to satisfy the aforementioned two properties; one can show that this non-stationarity is necessary but not sufficient for satisfying these properties; see [66] and later discussion. We will present the next two choices for this kernel, which cater to the two common scenarios for conglomerate multi-fidelity simulators from section 2.1.

We note that these kernel choices are only recommendations. The modeler should carefully consider prior domain knowledge to carefully select a kernel that captures such knowledge. With the kernel $K_{\mathbf{t}}$ specified (along with kernels $K_{\mathbf{x}}^{\delta}$ and $K_{\mathbf{x}}^{\phi}$), one can show that the response surface $\eta(\mathbf{x},\mathbf{t})$ follows a GP model with covariance function

$$K_{\eta}\{(\mathbf{x}_1,\mathbf{t}_1),(\mathbf{x}_2,\mathbf{t}_2)\} := \mathrm{Cov}[\eta(\mathbf{x_1},\mathbf{t}_1),\eta(\mathbf{x_2},\mathbf{t}_2)] = \sigma_1^2 K_{\mathbf{x}}^{\phi}(\mathbf{x}_1,\mathbf{x}_2) + \sigma_2^2 K_{\mathbf{x}}^{\delta}(\mathbf{x}_1,\mathbf{x}_2)K_{\mathbf{t}}^{\delta}(\mathbf{t}_1,\mathbf{t}_2). \tag{3.7}$$

The predictive equations for the CONFIG model then follow immediately from the standard GP equations (2.2) and (2.3) with kernel K_η given above and with the desired prediction point $(\mathbf{x}^*,\mathbf{0})$, as the goal is to predict the (limiting) highest-fidelity setting. We provide further details on these predictive equations in section 4.1.1.

3.1. Kernel option 1.

Consider the first kernel choice for $K_{\mathbf{t}}$ (Kernel 1), which we recommend for Scenario 1 above. This takes the non-stationary form

$$K_{\mathbf{t}}^{\delta}(\mathbf{t}_1,\mathbf{t}_2) = \exp\left\{-\sum_{r=1}^{q}\theta_r(t_{1,r}-t_{2,r})^2\right\} - \exp\left\{-\sum_{r=1}^{q}\theta_r t_{1,r}^2\right\} - \exp\left\{-\sum_{r=1}^{q}\theta_r t_{2,r}^2\right\} + 1. \tag{3.8}$$

Here, θ_r denotes the weight parameter for the rth fidelity parameter. A larger θ_r indicates greater sensitivity of discrepancy δ to the rth fidelity parameter, and vice versa. One can check that, with this kernel (3.8), the two desired properties (3.3) and (3.4) for δ are satisfied (see the supplementary materials (supp.pdf [local/web 301KB]), linked from the main article webpage), meaning that such a kernel indeed captures the aforementioned prior information on numerical convergence behavior.

Kernel 1 is inspired by the non-stationary covariance function in [22], which was proposed for a different task of uncertainty propagation for system outputs. The rationale for this kernel here is as follows. For simplicity, let $\delta(\mathbf{t})$ denote the bias term at some fixed input $\mathbf{x}$.

One way to ensure that $\delta(\mathbf{t})$ satisfies the limiting condition (3.3), i.e., $\lim_{\mathbf{t}\to\mathbf{0}}\delta(\mathbf{t})=0$, is to represent it as a difference of two terms

$$\delta(\mathbf{t})=\kappa(\mathbf{t})-\kappa(\mathbf{0}), \tag{3.9}$$

where $\kappa(\cdot)$ can be modeled as a GP. In words, the limiting condition on $\delta(\cdot)$ is enforced by centering κ by its response at the limiting fidelity $\mathbf{0}$. The covariance function for δ can then be written as

$$\begin{aligned}\mathrm{Cov}[\delta(\mathbf{t}_1),\delta(\mathbf{t}_2)] &= \mathrm{Cov}[\kappa(\mathbf{t}_1)-\kappa(\mathbf{0}),\kappa(\mathbf{t}_2)-\kappa(\mathbf{0})]\\ &= \mathrm{Cov}[\kappa(\mathbf{t}_1),\kappa(\mathbf{t}_2)]-\mathrm{Cov}[\kappa(\mathbf{t}_1),\kappa(\mathbf{0})]-\mathrm{Cov}[\kappa(\mathbf{t}_2),\kappa(\mathbf{0})]+\mathrm{Cov}[\kappa(\mathbf{0}),\kappa(\mathbf{0})].\end{aligned} \tag{3.10}$$

Kernel 1 in (3.8) can be recovered from (3.10) with a squared-exponential correlation function on κ and satisfies the desired limiting condition (3.3) by construction.

Kernel 1 has several appealing features for conglomerate multi-fidelity emulation. First, in many applications, one may have prior knowledge of the *continuity* of the underlying numerical solutions (e.g., from FEA theory). With Kernel 1, the corresponding prior process on discrepancy δ can be shown to yield continuous sample paths, thus capturing such prior knowledge from a Bayesian perspective. Second, the form of this kernel provides a flexible framework for modeling *interactions* between fidelity parameters across different stages. Compared to the additive structure in Kernel 2 introduced later, the latent GP model on $\kappa(\cdot)$ (with the squared-exponential kernel) provides a flexible framework for learning interactions between different fidelity parameters. Because of this, Kernel 1 appears to work best in Scenario 1 for emulating a *single* mechanism with multiple fidelity parameters, e.g., the FEA for beam deflection with different fidelities for each dimension, as such systems often have significant interaction effects between fidelity parameters, e.g., between mesh sizes of each dimension.

3.2. Kernel option 2.

Consider next the second choice for $K_\mathbf{t}$ (Kernel 2), which we recommend for the multi-stage *sequential* simulations in Scenario 2. This kernel takes the non-stationary form

$$K_\mathbf{t}^\delta(\mathbf{t}_1,\mathbf{t}_2)=\left[\sum_{r=1}^q \theta_r \min(t_{1,r},t_{2,r})^{\ell_r}\right]^\ell. \tag{3.11}$$

Here, θ_r is a weight parameter for the rth fidelity parameter, and ℓ_r and ℓ are positive integer kernel hyperparameters which we discuss later. Similar to Kernel 1, a greater θ_r allows for greater sensitivity of the discrepancy δ to the rth fidelity parameter. We can again show that with this kernel (3.11), the two properties (3.3) and (3.4) for bias δ are satisfied. This follows from the observations that $K_\mathbf{t}^\delta(\mathbf{t}',\mathbf{t}')$ tends to 0 as $\mathbf{t}'\to\mathbf{0}$ and that, given that some entries in $\mathbf{t}'$ are nonzero, we have $K_\mathbf{t}^\delta(\mathbf{t}',\mathbf{t}')\neq 0$ (see the supplementary materials (supp.pdf [local/web 301KB]) for further discussion). Such a kernel choice thus captures the desired prior information on numerical convergence. With Kernel 2, the resulting prior process on discrepancy δ can be viewed as a multi-variate extension of a standard Brownian motion model [14] and extends the non-stationary model (2.7) in [66], which tackled only the case of one fidelity parameter.

Kernel 2 has several appealing features for conglomerate multi-fidelity emulation, particularly when the multiple stages are performed *sequentially* over time (see Scenario 2 at the start of the section). One can show that the parametrization of this kernel is directly inspired by (and thus can capture prior information on) standard numerical convergence results for multi-stage simulators. To see why, consider first the simple setting of a *single* fidelity parameter t, and let v_0 and v_t be the exact and simulated solutions at fidelity t, respectively. In the case of finite element analysis (where t is the mesh grid size), it is well known [2] that the numerical error of the simulator can be upper bounded as

$$\|\nu_0 - \nu_t\| \leq C t^{\xi}, \tag{3.12}$$

where $||\cdot||$ is an appropriate norm on the solution space, ξ is a rate parameter, and C is a constant. In words, the numerical error resulting from mesh discretization decays polynomially as mesh size t decreases. Similar polynomial decay rates have also been shown for a broad range of fidelity parameters in numerical solvers, e.g., for elliptical PDEs [27] and large-eddy simulations in fluid mechanics [64].

Consider now the *multi-stage* simulators from Scenario 2, where a separate phenomenon is simulated sequentially at each stage. Suppose, at stage r, its precision is controlled by a fidelity parameter t_r. For this parameter t_r, further suppose the simulation error at this stage can be bounded by (3.12) with rate parameter ξ_r. One example of this is multi-stage finite element simulators when each stage involves a distinct finite element model (FEM) whose precision depends on a mesh size parameter t_r. Similar to before, let $\nu_{\mathbf{0}}$ and $\nu_{t_1,\cdots,t_q}$ denote the exact solution and the simulated solution at fidelity parameters $t_1,\ldots,t_q$. Applying the triangle inequality iteratively, the error between $\nu_{t_1,\ldots,t_q}$ and $\nu_{\mathbf{0}}$ can then be bounded as

$$\begin{aligned} ||v_{\mathbf{0}} - v_{t_1,\ldots,t_q}|| &\leq ||v_{\mathbf{0}} - v_{t_1,0,\ldots,0}|| + ||v_{t_1,0,\ldots,0} - v_{t_1,t_2,0,\ldots,0}|| + \cdots \\ &\quad + ||v_{t_1,\ldots,t_{q-1},0} - v_{t_1,\ldots,t_{q-1},t_q}|| \\ &\leq \sum_{r=1}^{q} C_r t_r^{\xi_r}, \end{aligned} \tag{3.13}$$

where $C_1,\ldots,C_q$ are again constants. We now show that Kernel 2 indeed captures the error bound (3.13) as *prior information* within its kernel specification. To see why, consider the prior standard deviation of the discrepancy term $\delta(\mathbf{x},\mathbf{t})$. From a Bayesian modeling perspective, this should capture the modeler's prior belief on the expected numerical error of the simulator. With $K_{\mathbf{t}}$ set as Kernel 2, one can show that this prior standard deviation takes the form

$$\sqrt{\mathrm{Var}\left\{\delta(\mathbf{x},\mathbf{t})\right\}} = \sigma_2 \left[\sum_{r=1}^{q} \theta_r t_r^{\ell_r}\right]^{\ell/2}. \tag{3.14}$$

Comparing (3.14) with (3.13), we see that they are precisely the same with the kernel hyperparameters set as $\ell = 2$ and $\ell_r = \xi_r$ for $r = 1,\ldots,q$. This suggests that with $K_{\mathbf{t}}$ chosen as Kernel 2, the resulting prior model on discrepancy $\delta(\mathbf{x},\mathbf{t})$ indeed captures (on expectation) the numerical error convergence of the multi-stage simulator.

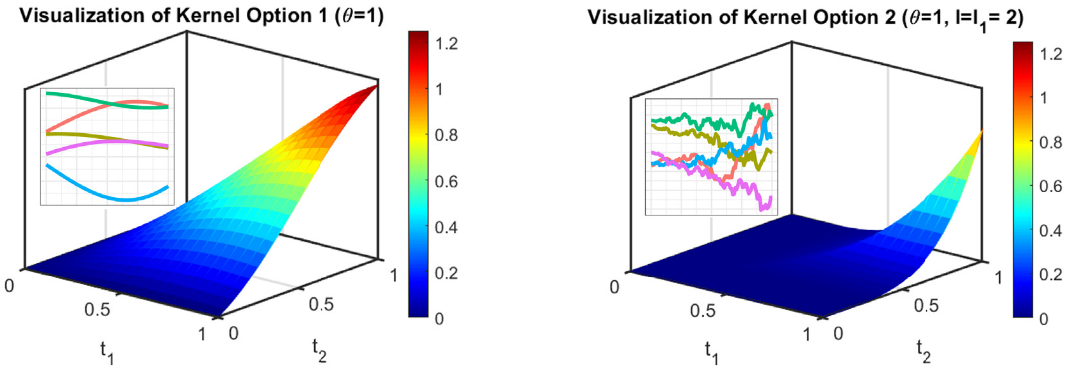
The above connection also helps guide how the specification of hyperparameters for Kernel 2. If the rate parameters $\xi_1,\ldots,\xi_q$ can be identified via a careful analysis of the error bound (3.12) at each stage, one can simply set the hyperparameters as $\ell_r = \xi_r$ for $r = 1,\ldots,q$.

However, for more complex multi-stage simulators, one may not be able to identify the precise error convergence rates at each stage. In such cases, the kernel hyperparameters can be estimated via maximum likelihood or a fully Bayesian approach (see section 4.1) or set at a fixed value (e.g., $\ell_r = \ell = 2$). Whether such hyperparameters are set a priori or inferred from data, the infusion of such prior information can yield noticeably improved predictive performance for multi-fidelity emulation, as we show later in section 6.2.

It is worth noting that, with the Brownian-like Kernel 2, sample paths from the discrepancy process $\delta(\mathbf{x}, \mathbf{t})$ will be highly nonsmooth, in the sense that within any neighborhood around $\mathbf{t} = \mathbf{0}$, the discrepancy $\delta(\mathbf{x}, \mathbf{t})$ will equal 0 an infinite number of times. This may be unintuitive for other properties of discretization error (see, e.g., equation (1) of [1]), which require that $\delta(\mathbf{x}, \mathbf{t}) = 0$ only when $\mathbf{t} = 0$. Our justification for Kernel 2 is not from such properties but rather from its ability to embed prior information on *expected* numerical convergence via its non-stationary specification. In applications where trajectory smoothness is a concern, Kernel 1 may be a better kernel choice; more on this is presented below.

Figure 3 visualizes the two proposed non-stationary kernels in the simple setting with a single fidelity parameter t. For Kernel 1, we set $\theta_q = 1$ and $q = 1$, and for Kernel 2, we set $\ell = \ell_q = 2$ and $q = 1$. We see these two kernels have noticeably different shapes: Kernel 1 shows a smooth and gradual increase as either t_1 or t_2 increases, whereas Kernel 2 exhibits a sharper increase and has a cusp along the line $t_1 = t_2$. This cusp causes the highly nonsmooth sample paths from Kernel 2, whereas the smoother kernel (Kernel 1) induces smoother sample paths, as can be seen from the corresponding sample paths in Figure 3.

3.3. Kernel recommendation. We provide next a concise summary of kernel recommendation for the CONFIG model. In applications where the conglomerate simulator uses multiple fidelity parameters (e.g., spatial mesh size or temporal discretization) for simulating a single mechanism (Scenario 1), we recommend the use of Kernel 1 (3.8), which can better account for stronger interactions between different fidelity parameters. We will encounter such an



(a) Visualizing Kernel 1 (3.8) with corresponding sample paths using parameters $q = 1$ and $\theta = 1$.

(b) Visualizing Kernel 2 (3.11) with corresponding sample paths using $q = 1$, $\theta = 1$, $\ell = \ell_1 = 2$.

Figure 3. *Visualization of both kernel options (with corresponding sample paths) for CONFIG with a single fidelity parameter t.*

application in section 6.1. On the other hand, in applications where the conglomerate simulator is comprised of multiple sequential stages that model for separate mechanisms, we recommend the use of Kernel 2 (3.11), which can be justified via numerical error analysis for these simulators. Our motivating QGP application falls within this setting, which we will investigate further in section 6.2.

There are, of course, applications that may not cleanly fall within the two presented scenarios; in such cases, careful consideration is needed for an informed kernel specification. In later numerical experiments, we have found that when there is little prior knowledge on the degree of interaction between fidelity parameters, Kernel 2 seems to be a considerably more robust choice for predictive modeling; we would thus recommend Kernel 2 for such problems.

4. Implementation.

We now discuss important implementation details for the CONFIG model. We present two parameter inference approaches, the first via maximum likelihood and the second via a fully Bayesian formulation for incorporating external knowledge and richer uncertainty quantification. We then outline plausible experimental design strategies.

4.1. Parameter inference.

4.1.1. Maximum likelihood.

We first present a maximum likelihood approach for estimating the CONFIG model parameters. Let $\boldsymbol{\Theta} = (\boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \sigma_1^2, \sigma_2^2)$ be the set of parameters to infer, where $\boldsymbol{\gamma}$ is the vector of weight parameters for $K_{\mathbf{x}}^{\phi}$, $\boldsymbol{\alpha}$ is the vector of weight parameters for $K_{\mathbf{x}}^{\delta}$, and $\boldsymbol{\theta}$ is the vector of weight parameters for the CONFIG kernel $K_{\mathbf{t}}$ (either Kernel 1 or Kernel 2). Here, we presume that the hyperparameters ℓ_r in Kernel 2 (3.11) are prespecified (similar to [66]) and thus not included in the parameter set $\boldsymbol{\Theta}$; in the setting where ℓ_r needs to be estimated, we can simply include them in $\boldsymbol{\Theta}$. Since $\eta(\mathbf{x}, \mathbf{t})$ can be expressed as a GP with the kernel given in (3.7), one can easily obtain an analytic expression for the likelihood function to optimize. More specifically, let the simulated multi-fidelity training data be $\mathbf{y} = (\eta(\mathbf{x}_i, \mathbf{t}_i))_{i=1}^n$, and let the matrix of basis functions for the GP mean be $\mathbf{F} = (\mathbf{f}(\mathbf{x}_1, \mathbf{t}_1)^T; \mathbf{f}(\mathbf{x}_2, \mathbf{t}_2)^T; \ldots; \mathbf{f}(\mathbf{x}_n, \mathbf{t}_n)^T)$ with corresponding coefficients $\boldsymbol{\beta}$. We thus aim to maximize the log-likelihood of the CONFIG model, given by

$$\max_{\boldsymbol{\Theta}} \left\{ -\frac{1}{2} \log \det \boldsymbol{\Sigma} - \frac{1}{2} (\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \mathbf{F}\boldsymbol{\beta}) \right\}, \tag{4.1}$$

where $\det \boldsymbol{\Sigma}$ is the determinant of the covariance matrix $\boldsymbol{\Sigma}$.

While the optimization problem (4.1) is quite high dimensional, standard nonlinear optimization algorithms, such as the L-BFGS-B method [46], appear to work well. One can further speed up this optimization procedure via an informed initialization of the parameters $\boldsymbol{\Theta}$. In particular, we have found that the correlation parameters $\boldsymbol{\alpha}$ can be well-initialized by first fitting a standard GP model with kernel $K_{\mathbf{x}}^{\delta}$ over the full training data (ignoring fidelity parameters). With these initial estimates, we then perform the L-BFGS-B nonlinear optimization, as implemented in the R package `stats` [4].

After maximum likelihood estimation, we would ideally like to integrate such estimates along with their uncertainties within the GP predictive equations (2.2) to predict the limiting highest-fidelity surface $\eta(\mathbf{x}^*, \mathbf{0})$ at a new input $\mathbf{x}^*$. However, this integration of uncertainty is difficult to do in closed form for all parameters (see [57]). We can, however, integrate estimation uncertainty on the mean coefficients $\boldsymbol{\beta}$ in an efficient manner. Following [1, 33], the CONFIG predictive mean and variance of $\eta(\mathbf{x}^*,\mathbf{0})$ with such uncertainty integrated (denoted as $\hat{\mu}_{\mathbf{0}}(\mathbf{x}^*)$ and $s^2_{\mathbf{0}}(\mathbf{x}^*)$, respectively) becomes

$$\begin{aligned}\hat{\mu}_{\mathbf{0}}(\mathbf{x}^*) &= \mu(\mathbf{x}^*,\mathbf{0}) + \mathbf{k}(\mathbf{x}^*,\mathcal{D})^T\mathbf{K}(\mathcal{D})^{-1}(\mathbf{y}_{\mathcal{D}} - \boldsymbol{\mu}(\mathcal{D})),\\ s^2_{\mathbf{0}}(\mathbf{x}^*) &= k(\mathbf{x}^*,\mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*,\mathcal{D})^T\mathbf{K}(\mathcal{D})^{-1}\mathbf{k}(\mathbf{x}^*,\mathcal{D}) + (\mathbf{f}(\mathbf{x}^*,\mathbf{0}) - \mathbf{k}(\mathbf{x}^*,\mathcal{D})^T\mathbf{K}(\mathcal{D})^{-1}\mathbf{F})^T\\ &\quad (\mathbf{F}^T\mathbf{K}(\mathcal{D})^{-1}\mathbf{F})^{-1}(\mathbf{f}(\mathbf{x}^*,\mathbf{0}) - \mathbf{k}(\mathbf{x}^*,\mathcal{D})^T\mathbf{K}(\mathcal{D})^{-1}\mathbf{F}).\end{aligned} \tag{4.2}$$

Unknown model parameters in (4.2) can then be plugged in via the maximum likelihood estimates (4.1). With this, we can then construct the 95% predictive interval on the limiting highest-fidelity output $\eta(\mathbf{x}^*,\mathbf{0})$ as

$$\left(\hat{\mu}_{\mathbf{0}}(\mathbf{x}^*) - 1.96\sqrt{s^2_{\mathbf{0}}(\mathbf{x}^*)},\ \hat{\mu}_{\mathbf{0}}(\mathbf{x}^*) + 1.96\sqrt{s^2_{\mathbf{0}}(\mathbf{x}^*)}\right). \tag{4.3}$$

4.1.2. Fully Bayesian inference. In situations where a richer quantification of uncertainty is desired, a fully Bayesian approach to parameter inference may be appropriate. Below, we present one such approach for the CONFIG model which leverages a Metropolis-within-Gibbs algorithm [19] for posterior sampling. For an easier derivation of the full conditional distributions, we consider a reparametrization of the covariance kernel (3.7) for $\eta(\mathbf{x},\mathbf{t})$ as

$$K_\eta\{(\mathbf{x}_1,\mathbf{t}_1),(\mathbf{x}_2,\mathbf{t}_2)\} = \sigma^2\left\{K^\phi_{\mathbf{x}}(\mathbf{x}_1,\mathbf{x}_2) + \lambda K^\delta_{\mathbf{x}}(\mathbf{x}_1,\mathbf{x}_2)K^\delta_{\mathbf{t}}(\mathbf{t}_1,\mathbf{t}_2)\right\}, \tag{4.4}$$

where $\sigma^2 := \sigma_1^2$ and $\lambda := \sigma_2^2/\sigma_1^2$. Here, the new parameter λ captures the degree of nonstationarity in the kernel from the influence of the fidelity parameters $\mathbf{t}$. When $\lambda = 0$, the covariance kernel becomes a stationary kernel that depends on only input parameters $\mathbf{x}$.

With this reparametrization, the parameter set to infer is given by $\tilde{\boldsymbol{\Theta}} = (\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\alpha},\boldsymbol{\theta},\sigma^2,\lambda)$. It is straightforward to show that

$$\mathbf{y}|\tilde{\boldsymbol{\Theta}} \sim \mathcal{N}(\mathbf{F}\boldsymbol{\beta},\boldsymbol{\Sigma}), \tag{4.5}$$

using the same notation as in (4.1). Table 1 summarizes the priors assigned on parameters $\tilde{\boldsymbol{\Theta}}$. As before, this specification does not include ℓ_r for Kernel 2, but one can always leverage a reasonable prior distribution on ℓ_r if information is not known on such parameters. Here, the prior hyperparameters can either be set via prior information or set in a weakly informative fashion with $a_\lambda = b_\lambda = 1$ and $a = b = 0.001$ for the remaining hyperparameters.

With the priors specified, we now proceed to the posterior sampling algorithm. Of the model parameters in $\tilde{\boldsymbol{\Theta}}$, we can derive full conditional distributions for two parameters, $\boldsymbol{\beta}$ and $1/\sigma^2$:

$$\boldsymbol{\beta}|\mathbf{y},\boldsymbol{\gamma},\boldsymbol{\alpha},\boldsymbol{\theta},\sigma^2,\lambda \sim \mathcal{N}((\mathbf{F}^T\boldsymbol{\Sigma}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\boldsymbol{\Sigma}^{-1}\mathbf{y},\sigma^2(\mathbf{F}^T\boldsymbol{\Sigma}^{-1}\mathbf{F})^{-1}), \tag{4.6}$$

$$1/\sigma^2|\mathbf{y},\boldsymbol{\gamma},\boldsymbol{\alpha},\boldsymbol{\theta},\lambda,\boldsymbol{\beta} \sim \text{Gamma}\left(a_\sigma + \frac{n}{2},(1+\lambda)b_\sigma + \frac{1}{2}(\mathbf{y}-\mathbf{F}\boldsymbol{\beta})^T\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\mathbf{F}\boldsymbol{\beta})\right). \tag{4.7}$$

For the remaining parameters in $\tilde{\boldsymbol{\Theta}}$, we make use of Metropolis–Hastings [40] steps for sampling the full conditional distributions, as implemented in the R package `MHadaptive` [8]. We then

Table 1
Hierarchical model specification for the fully Bayesian CONFIG model.

Model	Prior specification
CONFIG: $\eta(\mathbf{x},\mathbf{t}) \sim \mathcal{GP}\{\boldsymbol{F\beta}, K_\eta(\cdot,\cdot)\}$	$[\beta_1,\beta_2,\ldots,\beta_m] \overset{\text{i.i.d.}}{\sim} 1$
Priors: $[\tilde{\boldsymbol{\Theta}}] = [\boldsymbol{\beta}][\lambda][\sigma^2\|\lambda][\boldsymbol{\gamma}][\boldsymbol{\alpha}][\boldsymbol{\theta}]$	
Non-stationary parameter	$\lambda \sim \text{Beta}(a_\lambda, b_\lambda)$
Kernel precision	$1/\sigma^2\|\lambda \sim \text{Gamma}(a_\sigma,(1+\lambda)b_\sigma)$
Weight parameters	$\gamma_1,\gamma_2,\ldots,\gamma_p \overset{\text{i.i.d.}}{\sim} \text{Gamma}(a_\gamma,b_\gamma)$
Weight parameters	$\alpha_1,\alpha_2,\ldots,\alpha_p \overset{\text{i.i.d.}}{\sim} \text{Gamma}(a_\alpha,b_\alpha)$
CONFIG weight parameters	$\begin{cases}\theta_1,\theta_2,\ldots,\theta_q \overset{\text{i.i.d.}}{\sim} \text{Gamma}(a_\theta,b_\theta) \text{ for Kernel 1}\\ \theta_1,\theta_2,\ldots,\theta_q \overset{\text{i.i.d.}}{\sim} \text{Beta}(a_\theta,b_\theta) \text{ for Kernel 2}\end{cases}$

Algorithm 4.1. Metropolis-within-Gibbs sampler for the CONFIG model.

```
Input: Training data $\{\mathbf{x}_i,\mathbf{t}_i\}_{i=1}^n$, $\mathbf{y}=(\eta(\mathbf{x}_i,\mathbf{t}_i))_{i=1}^n$; testing input $\mathbf{x}^*$; prior hyperparameters $a_\lambda$, $b_\lambda$, $a_\sigma$, $b_\sigma$, $a_\gamma$, $b_\gamma$, $a_\alpha$, $b_\alpha$, $a_\theta$, $b_\theta$; number of desired MCMC draws $M$; burn-in period $M_{\text{burn-in}}$ and thinning rate $T$.
Output: Draws from the posterior distribution $[\tilde{\boldsymbol{\Theta}}|\mathbf{y}]$.
Initialize the parameters $\tilde{\boldsymbol{\Theta}}^{[0]}$ from the prior.
for iter $= 1,\ldots,M_{\text{burn-in}} + TM$ do
    Draw $\boldsymbol{\beta}^{[\text{iter}]}$ from the full conditional distribution (4.6).
    Draw $1/\sigma^{2[\text{iter}]}$ from the full conditional distribution (4.7).
    For the remaining parameters $\{\boldsymbol{\gamma},\boldsymbol{\alpha},\boldsymbol{\theta},\boldsymbol{\lambda}\}$, perform one step of Metropolis-within-Gibbs sampling using parameters $\{\boldsymbol{\beta}^{[\text{iter}]}, 1/\sigma^{2[\text{iter}]}\}$.
end for
Discard the first $M_{\text{burn}}$ draws and thin the remaining draws at a rate of $T$ to obtain draws $\{\tilde{\boldsymbol{\Theta}}^{[m]}\}_{m=1}^M$.
Return: Draws $\{\tilde{\boldsymbol{\Theta}}^{[m]}\}_{m=1}^M$ from the posterior distribution $[\tilde{\boldsymbol{\Theta}}|\mathbf{y}]$.
```

iterate these full conditional sampling steps within a Gibbs sampler for posterior exploration of $[\tilde{\boldsymbol{\Theta}}|\mathbf{y}]$. Algorithm 4.1 presents the detailed steps for this Metropolis-within-Gibbs sampler for the CONFIG model with details on burn-in and thinning.

Finally, with the posterior draws $\{\tilde{\boldsymbol{\Theta}}^{[m]}\}_{m=1}^M$ obtained from Algorithm 4.1, we can easily estimate the posterior predictive mean at a new test point $\mathbf{x}^*$ by marginalizing over $\tilde{\boldsymbol{\Theta}}$:

$$\mathbb{E}[\eta(\mathbf{x}^*,\mathbf{0})|\mathbf{y}] \approx \frac{1}{M}\sum_{m=1}^{M} \hat{\eta}\left(\mathbf{x}^*,\mathbf{0}|\tilde{\boldsymbol{\Theta}}^{[m]}\right),$$

where $\hat{\eta}(\mathbf{x}^*,\mathbf{0}|\tilde{\boldsymbol{\Theta}}^{[m]})$ is the closed-form GP predictive mean in (2.2) with fixed hyperparameters $\tilde{\boldsymbol{\Theta}}^{[m]}$. This serves as the emulator for the fully Bayesian CONFIG model. One can also quantify its uncertainty via posterior predictive draws on $\eta(\mathbf{x}^*,\mathbf{0})|\mathbf{y}$. These can be obtained by sampling a batch of draws from the predictive distribution $[\eta(\mathbf{x}^*,\mathbf{0})|\mathbf{y},\tilde{\boldsymbol{\Theta}}^{[m]}]$ in (2.2) given parameters $\tilde{\boldsymbol{\Theta}}^{[m]}$ and then repeating this procedure on all posterior draws $\{\tilde{\boldsymbol{\Theta}}^{[m]}\}_{m=1}^M$.

4.2. Experimental design. Of course, given a fixed and limited computational budget, an experimenter would want to maximize the predictive power of the fitted multi-fidelity emulator model. For GP models, space-filling designs [31]—which aim to uniformly fill up the design space—are commonly used and have desirable information-theoretic and predictive properties [30]. Different notions of space-filling designs have been explored in the literature, including maximin designs [30, 42], minimax designs [30, 38], and maximum projection (MaxPro) designs [32].

For the CONFIG model, there are several ways in which one can adapt existing space-filling designing methods. One approach is to (i) adopt a space-filling design over the combined design space of both input parameters $\mathbf{x}$ and fidelity parameters $\mathbf{t}$. Such a design ensures that training points are not only well-spaced out over the input space for prediction at untested settings but also well-spaced out over the fidelity space to better learn the effects of individual fidelity parameters. Another approach might be (ii) a crossed array design [69] between input and fidelity space, which are popular designs for robust parameter design. In such a design, one first generates two space-filling designs, one over the input space and the other over the fidelity space, and then takes for the final design all combinations of input and output points. Both designs appear to yield good performance: the designs in (i) are used for our numerical experiments and cantilever beam deflection application, and the designs in (ii) are used for the emulation of the QGP evolution. The problem of optimal experimental design for the proposed non-stationary CONFIG model is quite intriguing, and we aim to pursue this in future work.

5. Numerical experiments.

We now explore the performance of the proposed CONFIG model in a suite of simulation experiments with multiple fidelity parameters. We compare the CONFIG model with several existing emulator models. The first model is a standard GP emulator with a squared-exponential correlation function on both input parameters $\mathbf{x}$ and fidelity parameters $\mathbf{t}$; one then uses the fitted model to predict at $\mathbf{t} = \mathbf{0}$. We call this model simply the "standard GP" emulator. The second model is the TWY model [66], which uses a single fidelity parameter. Since there are multiple fidelity parameters in the target conglomerate problem, we will first compute the arithmetic or geometric mean of the fidelity parameters $t_1, \ldots, t_q$ then apply the TWY model with this single aggregate fidelity parameter. We call the resulting models the TWY (ARITH) and TWY (GEOM) emulators, respectively.

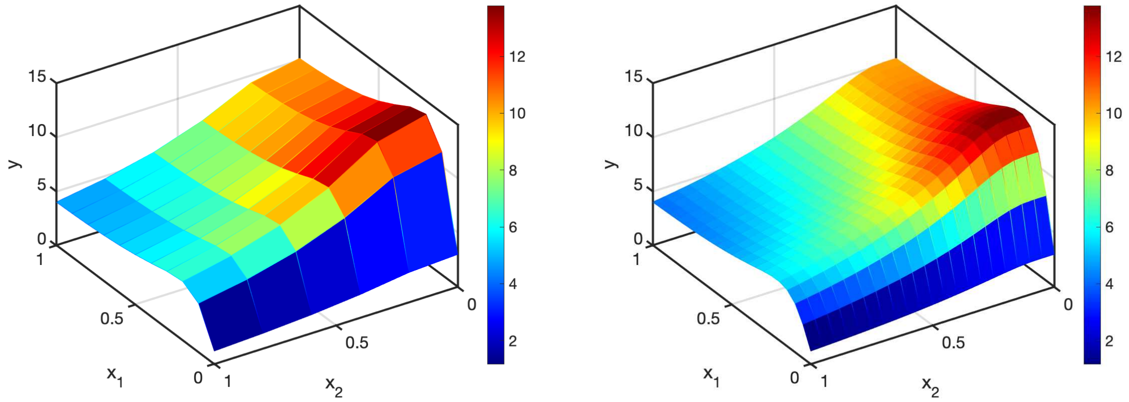
In the following, we investigate the performance of these models on multi-fidelity extensions of two test functions, the 2D Currin function [11] and the 4D Park function [9]. For CONFIG, we follow section 4 and set the power parameters in Kernel 2 as $\ell_r = \ell = 2$. Kernel hyperparameters for our model are estimated via maximum likelihood in sections 5.1 and 5.2, and its fully Bayesian counterpart is explored in section 5.4.

5.1. Multi-fidelity Currin function.

Our first test function builds off of the 2D Currin test function in [11]:

$$\phi(\mathbf{x}) = \left[1 - \exp\left(-\frac{1}{2x_2}\right)\right] \frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}, \tag{5.1}$$

where $\mathbf{x} = [x_1, x_2] \in [0,1]^2$. We then build a lower-fidelity representation of this function, denoted as $\eta(\mathbf{x}, \mathbf{t})$, with two fidelity parameters $\mathbf{t} = [t_1, t_2]$ via piecewise grid interpolation. More



(a) $\eta(\mathbf{x},\mathbf{t})$ with $t_1 = 0.1$ and $t_2 = 0.2$.

(b) $\eta(\mathbf{x},\mathbf{t})$ with $t_1 = 0.05$ and $t_2 = 0.05$.

Figure 4. *Visualization of the multi-fidelity Currin function at two different fidelity settings.*

specifically, this approximation is carried out in two steps. First, we generate a rectangular grid in the input space, where the dimension of each mesh cell is $t_1 \times t_2$. Next, we evaluate the underlying function (5.1) at the mesh grid points and perform piecewise grid interpolation to construct a lower-fidelity version of (5.1). This procedure is effectively the same as finite element meshing, which splits the input domain into many smaller elements. Figure 4 visualizes this test function $\eta(\mathbf{x},\mathbf{t})$ with $(t_1,t_2) = (0.1, 0.2)$ and $(0.05, 0.05)$. It is clear that as t_1 and t_2 become smaller, $\eta(\mathbf{x},\mathbf{t})$ becomes closer to the underlying Currin function (5.1), which is as desired.

We then compare the CONFIG emulator with the aforementioned baseline models. For each experiment, we first generate $n = 50$ design points over *both* input and fidelity parameters via the MaxPro design [32]. Here, we set the range for each fidelity parameter to be between 0.1 and 0.4 to mimic the reality that simulations are prohibitively expensive for small choices of fidelity parameters t_i. Using this design, we then collect training data from the multi-fidelity Currin function $\eta(\mathbf{x},\mathbf{t})$. For validation, we randomly select $N = 1{,}000$ points over the input parameter space as the testing set and compare how well these models predict the Currin function $\phi(\mathbf{x})$ in terms of mean squared error (MSE) and its empirical coverage ratio of 95% predictive intervals. This procedure is then replicated 20 times.

Figure 5(left) shows the boxplots of testing MSEs and a scatterplot of empirical coverage rates against MSEs for the compared models, with Table 2(left) reporting its corresponding average MSEs and coverage ratios. There are several observations of interest. First, CONFIG (with either Kernel 1 or Kernel 2) outperforms existing models in terms of average MSE. This suggests that, by embedding the underlying conglomerate multi-fidelity structure within the non-stationary kernel specification, the proposed model can indeed provide better emulation over models that do not explicitly integrate this information. The improved performance of CONFIG over the TWY models also suggests that, when *multiple* fidelity parameters are present in the simulator, the use of such information can be useful for reducing emulation error over the TWY models, which aggregate fidelity into a single parameter. Finally, CONFIG

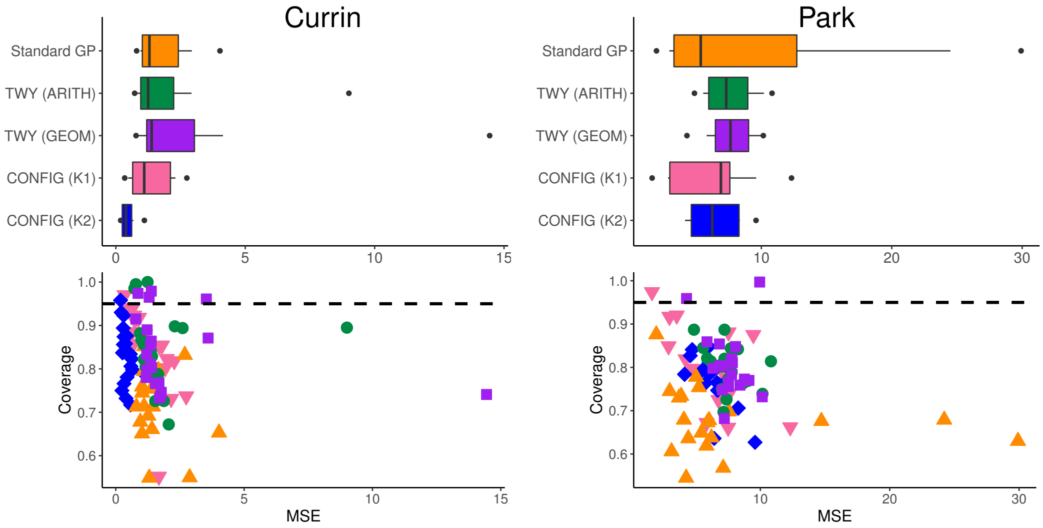


Figure 5. *(Top) Boxplots of MSEs for the multi-fidelity Currin and multi-fidelity Park experiments. (Bottom) Scatterplots of empirical coverage ratios vs. MSEs for the multi-fidelity Currin and multi-fidelity Park experiments. Each dot represents a replication of the experiment, and the black dashed lines denote the nominal* 95% *rate.*

Table 2

Average testing MSEs and empirical coverage ratios for the multi-fidelity Currin and Park experiments over 20 *replications.*

Model	Avg. MSE (Currin)	Avg. coverage (Currin)	Avg. MSE (Park)	Avg. coverage (Park)
Standard GP	1.537	71.93%	7.587	67.96%
TWY (ARITH)	1.772	84.46%	7.509	79.96%
TWY (GEOM)	2.233	83.96%	7.609	**80.68%**
CONFIG (Kernel 1)	1.310	**84.90%**	**6.141**	80.60%
CONFIG (Kernel 2)	**0.438**	83.18%	6.429	76.41%

with Kernel 2 provides noticeably better performance than Kernel 1; this may be because there is little interaction between the two fidelity parameters under the piecewise grid interpolation of $\eta(\mathbf{x},\mathbf{t})$.

As for coverage ratios (Figure 5 and Table 2(left)), we first note that among the 20 replications, not all empirical coverage ratios can attain the nominal rate of 95%. This highlights an inherent challenge for multi-fidelity uncertainty quantification, as one requires *extrapolation* beyond the range of simulated fidelity parameters to predict for the (limiting) highest-fidelity code. This becomes more pronounced for the considered conglomerate setting with *multiple* fidelity parameters and limited data. We see that the standard GP yields the lowest coverage rates; this is unsurprising, as it fails to account for the non-stationary behavior of fidelity parameters $\mathbf{t}$ from numerical convergence. Both CONFIG and TWY provide comparable coverage rates that are slightly below 95%, particularly that for CONFIG Kernel 2. We will address this undercoverage later in section 5.4.

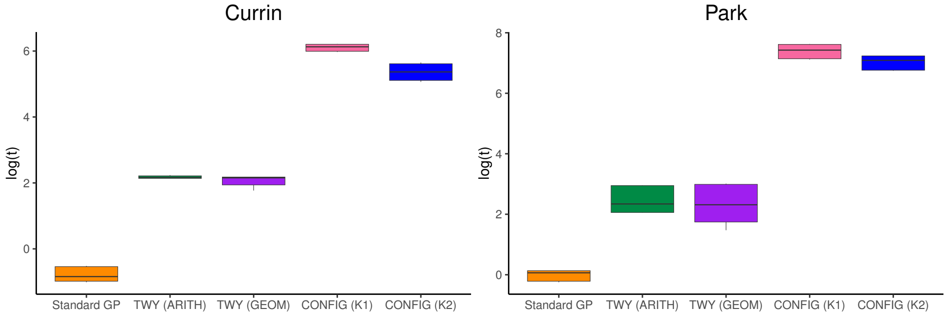


Figure 6. *Boxplots of computational times (in log-seconds) for the multi-fidelity Currin and Park experiments over* 20 *replications.*

Figure 6 shows the computational times of the compared methods over 20 replications, where all experiments are run on an 16-core CPU and 64 Gb memory. Not surprisingly, the two CONFIG models have higher computational costs, as it requires the estimation of more parameters within a non-stationary GP framework. Such costs, however, are considerably less than the computational resources required to simulate training data from expensive computer experiments in practice.

5.2. Multi-fidelity Park function. Our second test function builds off of the 4D Park test function in [9]:

$$\phi(\mathbf{x}) = \frac{x_1}{2}\left[\sqrt{1+(x_2+x_3^2)\frac{x_4}{x_1^2}}-1\right] + (x_1+3x_4)\exp\left(1+\sin(x_3)\right). \tag{5.2}$$

We again build a lower-fidelity representation of $\phi(\mathbf{x})$, denoted as $\eta(\mathbf{x},\mathbf{t})$, using four fidelity parameters $\mathbf{t} = (t_1,\ldots,t_4)$ via piecewise grid interpolation. Similar to before, we use MaxPro designs (with $n = 50$ design points) over both input and fidelity parameters, with a range of $[0.2, 0.5]$ for each fidelity parameter. The same emulator models are compared as before, and the experiment is replicated 20 times over $N = 1{,}000$ random testing points.

Figure 5(right) shows the boxplots of testing MSEs and a scatterplot of empirical coverage rates against MSEs for the compared models, with Table 2(right) reporting its average MSEs and coverage ratios. We see again that the proposed CONFIG model outperforms its competitors by a noticeable margin in terms of MSE, which affirms the value of embedding prior information on the conglomerate multi-fidelity simulator within a carefully constructed non-stationary kernel. For coverage ratios, all five models yield lower coverage rates than for the Currin function; this may be due to the increasing challenge of uncertainty quantification for extrapolation in a higher-dimensional setting. Despite this, CONFIG (with Kernels 1 and 2) maintains comparable coverage to TWY and higher coverage than the standard GP.

5.3. Modified multi-fidelity Currin function. We now explore a more complex modification of the Currin function, which integrates additional structure for the high-fidelity function. Our third test function $\eta_{\mathrm{mod}}(\mathbf{x},\mathbf{t})$ takes the form

$$\eta_{\mathrm{mod}}(\mathbf{x},\mathbf{t}) = \eta_{\mathrm{currin}}(\mathbf{x},\mathbf{t}) + (1-t_1)\sin(x_1) + (1-t_2)\cos(x_2), \tag{5.3}$$

Table 3

Average testing MSEs and empirical coverage ratios for the modified multi-fidelity Currin experiment over 20 *replications.*

Model	Avg. MSE (modified Currin)	Avg. coverage (modified Currin)
Standard GP	2.014	64.65%
TWY (ARITH)	2.132	**85.48%**
TWY (GEOM)	2.073	83.02%
CONFIG (Kernel 1)	1.559	84.36%
CONFIG (Kernel 2)	**0.632**	73.12%

where $\eta_{\text{currin}}(\mathbf{x},\mathbf{t})$ is the multi-fidelity Currin function used in section 5.1. The two additional terms $\sin(x_1)$ and $\cos(x_2)$ impose structure in the high-fidelity function (i.e., with $t_1=t_2=0$), which gets blurred out at lower fidelities (i.e., as t_1 or t_2 increases). This reflects scenarios where higher-fidelity refinements of the computer code may reveal additional structure not captured at lower fidelities. The simulation setup used here is the same as in section 5.1, with the visualizations provided in the supplementary materials (supp.pdf [local/web 301KB]).

Table 3 shows the average MSEs and coverage ratios over 20 replications. We see again that the CONFIG (with either Kernel 1 or 2) outperforms its competitors in terms of testing MSE. For coverage ratios, we see that while the TWY with arithmetic mean yields the highest coverage ratio, this model also returns the highest predictive error, which is clearly not desirable. The CONFIG with Kernel 1 achieves a comparably high coverage ratio with noticeably less prediction error, and the CONFIG with Kernel 2 yields a much lower prediction error at the cost of slight undercoverage (this can be addressed via the following fully Bayesian implementation).

5.4. Fully Bayesian implementation.

One reason for the slight undercoverage of CONFIG in previous experiments is that it employs plug-in parameters estimated via maximum likelihood (section 4.1.1). One solution is to employ a fully Bayesian implementation of the model (section 4.1.2); we explore its performance for the earlier Currin and Park experiments. For the MLE approach, its 95% predictive interval is obtained from the closed-form distribution (4.2) with plug-in parameter estimates. For the fully Bayesian approach, we use its 95% highest-posterior-density interval computed from posterior draws on the predictive distribution $[\eta(\mathbf{x}^*,\mathbf{0})|\mathbf{y}]$. These draws are obtained via five parallel MCMC chains from Algorithm 4.1 with random initialization. Each chain was run for 10,000 iterations, with the first 5,000 discarded as burn-in and the remaining draws thinned by a factor of 50 to reduce autocorrelation. Since the fully Bayesian model is more costly to fit, we demonstrate this on only one set of training/testing data from earlier experiments.

For MCMC, convergence was assessed via the Gelman–Rubin statistic [20], as implemented in the R package `coda` [51]. For the fully Bayesian CONFIG model with Kernel 2, all model parameters have a Gelman–Rubin statistic below 1.2, thus suggesting the MCMC has converged [3]. However, for the fully Bayesian CONFIG model with Kernel 1, we encountered very poor mixing performance and numerical instability issues, as the posterior distribution of kernel hyperparameters appears to be highly complex and multi-modal. Since undercoverage seems to be less pronounced for Kernel 1 (see Table 2), we thus recommend a fully Bayesian

Table 4

Testing MSEs and empirical coverage ratios for the plug-in MLE and fully Bayesian CONFIG model with Kernel 2 in the two multi-fidelity Currin and Park experiments with the first set of training/testing data.

Model	MSE (Currin)	Coverage (Currin)	MSE (Park)	Coverage (Park)	MSE (Mod Currin)	Coverage (Mod Currin)
CONFIG (Kernel 2, MLE)	0.621	82.30%	6.041	78.40%	0.706	82.80%
CONFIG (Kernel 2, Bayesian)	0.615	86.70%	6.396	83.40%	0.496	93.50%

implementation for only Kernel 2 to address the aforementioned undercoverage issue with plug-in MLEs.

With this in mind, Table 4 summarizes the MSEs and coverage ratios for the plug-in MLE and fully Bayesian CONFIG model using Kernel 2. We see that, while the MSEs of the plug-in MLE approach are quite small, its coverage ratios (82.30% and 78.40% for the Currin and Park functions, respectively) are lower than the desired rate of 95%. This is again unsurprising since plug-in MLEs do not account for parameter estimation uncertainty. The fully Bayesian CONFIG model yields similarly small MSEs but provides noticeably closer coverage to the desired 95% rate by factoring in posterior uncertainty on parameters. While this still yields slight undercoverage (which is unsurprising since extrapolation with GPs is inherently difficult, particularly with multiple fidelity parameters), we see that a fully Bayesian implementation can indeed provide improved uncertainty quantification for conglomerate multi-fidelity emulation.

6. Applications.

Finally, we explore the usefulness of the proposed model in two applications. The first application involves the conglomerate multi-fidelity emulation of a cantilever beam deflecting under stress. The second application is the earlier motivating problem of multi-stage multi-fidelity emulation of the quark-gluon plasma produced in heavy-ion collisions. In both applications, parameter estimation for CONFIG is performed via maximum likelihood (see section 4.1.1).

6.1. Cantilever beam deflection.

The first application investigates the static stress analysis on a cantilever beam. Beam structures are commonly used in finite elements to model transverse loads and deformation under various circumstances, and the study of their deflection behavior is a canonical problem in FEA and has been studied extensively [6, 25, 45]. Here, we use it to evaluate the performance of our modeling framework. Figure 7 shows an illustration of the beam for our study, where one end surface of the beam is fixed, and an external pressure field is applied on its top surface. The deflection of this beam under stress is typically simulated using FEA simulations, which can be computationally expensive. For our experiments, these FEA simulations are carried out using the ABAQUS software [59] with rectangular mesh cells.

The setup is as follows. The beam dimensions are specified by its breadth d_1, its height d_2, and its length d_3. We further let $d_1 = d_2$ so the cross-section of the beam is square-shaped (see Figure 7). We then set Young's modulus of the beam (which parametrizes the stiffness of the beam) to be 200 MPa and the Poisson ratio (which measures the deformation of the beam under loading) to be 0.28, with material properties corresponding to steel. For the external

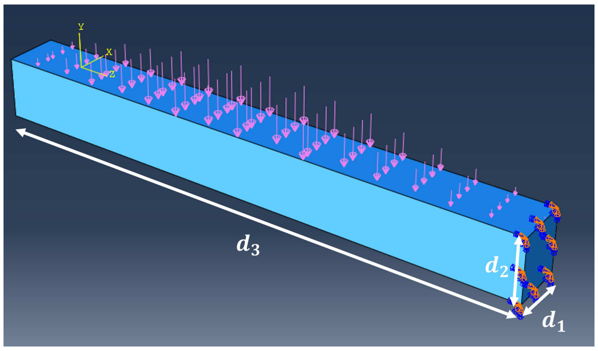


Figure 7. *Beam cantilever simulation with fixed end surface as shown in ABAQUS.*

pressure field, which is applied vertically downward on top of the beam, we employed the continuous half-sine pressure field given by

$$p(w) = C_1 x_1 \sin(C_2 w/x_2). \tag{6.1}$$

Here, $w \in [0, d_3]$ denotes the location along the beam from the fixed end, $C_1 = 2000$ and $C_2 = \pi/200$ are constants, x_1 is a scale factor for the pressure, and x_2 parametrizes the length of the beam, i.e., $d_3 = 200x_2$. An additional input parameter x_3 controls the width and breadth of the beam cross-section $x_3 = 20d_1 = 20d_2$. There are thus a total of three input parameters $\mathbf{x} = [x_1, x_2, x_3] \in [0,1]^3$ for this study.

For fidelity parameters, it is natural to consider a meshing procedure that partitions the beam into smaller 3D mesh rectangles. The size of these mesh rectangles can be controlled by three fidelity parameters, which dictate the size of the mesh rectangles in each dimension. In other words, the three fidelity parameters $t_1, t_2, t_3 \in (0,1)$ determine the *scale* of the finite elements. As a result of the pressure field, the cantilever beam will deflect downward, resulting in deflection at its tip. The response of interest is taken to be the amount of tip deflection. The goal is thus to train an emulator model which, using a carefully designed training set of simulation runs over different inputs $\mathbf{x}$ and fidelities $\mathbf{t}$, efficiently predicts the "exact" solution for tip deflection (i.e., at $\mathbf{t} = \mathbf{0}$) of a new beam with inputs $\mathbf{x}$.

The experiment is carried out as follows. To generate training data, we first run the simulator (ABAQUS) on a $n = 50$-point MaxPro design [32] over the combined space of input parameters $\mathbf{x}$ and fidelity parameters $\mathbf{t}$, which required about 4.5 hours of computation. For fidelity parameters, we set it to be $\mathbf{t} \in [1/31, 1/3]^3$, which ensures we have an integer number of finite elements at the edge case in each dimension (for $\mathbf{t}$ values in between, we round up to the nearest integer). For validation, we further run the simulator on 30 new cantilever configurations (the testing set) where each takes about 1 hour, uniformly sampled over the input space, to test the performance of each model (in terms of MSE and coverage) in predicting the tip deflections. While the "exact" response with $\mathbf{t} = \mathbf{0}$ cannot be obtained numerically, this can be well-approximated by running the simulator at very fine mesh sizes; in our case, we used $\mathbf{t} = [0.025, 0.025, 0.005]$ for testing points, which provided a sufficiently fine mesh according to a mesh validation study. One simulation run at this high-fidelity setting requires around 1 hour of computation, meaning there is a considerable opportunity for a multi-fidelity emulator to greatly speed up design exploration.

The same emulators as before (the standard GP, the two TWY models, and the two CONFIG models) are used for comparison. Here, we recommend the use of Kernel 1 for CONFIG, as the application involves the simulation of a single mechanism with multiple fidelity parameters (Scenario 1); Kernel 2 is, however, included for comparison. We further set $\ell = \ell_r = 2$ for CONFIG and $\ell = 2$ for the two TWY models; such a choice captures the fact that the governing deflection equation (between beam deflection and span) involves the derivative of the deflection [66]. In addition, we include a "high-fidelity GP" emulator model, which is trained only on data from the high-fidelity simulator with $\mathbf{t} = [0.025, 0.025, 0.005]$. For a fair comparison, this model is trained on high-fidelity points from a four-point MaxPro design, which requires comparable time to simulate as the earlier 50-point designs over the combined input-fidelity space.

Table 5 summarizes the MSEs, average standard errors, and empirical coverage ratios (of 95% predictive intervals) for the compared emulators over 30 test points. Again, we see that CONFIG with the recommended Kernel 1 (but also with Kernel 2) yields noticeably improved predictive performance over existing methods. This again highlights the advantage of an informed kernel specification from the conglomerate multi-fidelity simulator, and is particularly apparent in the cantilever beam application, where the three fidelity parameters bear physical importance. For beam bending, the accuracy of simulations is known to be more sensitive to the mesh density along the beam span (i.e., d_3) [10]. By explicitly modeling this conglomerate multi-fidelity structure, CONFIG can identify the greater importance of this fidelity parameter via inference of its weight parameter, thus allowing for improved predictions over existing models that ignore such structure.

In terms of coverage ratios, the 95% predictive intervals for both CONFIG models cover 26 out of 30 test points. Although this is slightly lower than the desired 95%, this is in line with earlier numerical experiments and may be due to the inherent challenge of multi-dimensional extrapolation with GPs. While the high-fidelity GP and TWY (ARITH) models have higher coverage ratios than CONFIG, their predictive uncertainties (measured by average standard error) are significantly larger (28.28 and 16.89, respectively) compared to the CONFIG models (2.75 and 4.77). These large uncertainties, along with poor predictions, make such models unappealing despite their high coverage ratios. The proposed CONFIG models provide markedly better predictions with lower uncertainties while maintaining comparable coverage ratios to existing models.

Further insight can be gleaned by comparing the performance of CONFIG with Kernel 1 vs. Kernel 2. Here, Kernel 1 provides slightly better predictions compared to Kernel 2. This

Table 5
Result comparison for the beam deflection application.

Model	MSE	Average standard error	Coverage ratio
High-fidelity GP	475.50	28.28	**30/30**
Standard GP	211.13	8.43	25/30
TWY (ARITH)	747.30	16.89	27/30
TWY (GEOM)	551.07	11.29	26/30
CONFIG (Kernel 1)	**23.58**	**2.75**	26/30
CONFIG (Kernel 2)	33.18	4.77	26/30

is not too surprising since this cantilever beam deflection can be viewed as a single-mechanism multi-fidelity problem and can thus be classified under Scenario 1 (see section 3), where the experiment simulates a *single* mechanism with multiple fidelity parameters. Kernel 1 appears to be better suited at capturing the more complex interactions between fidelity parameters, thus leading to slightly better performance than Kernel 2.

6.2. Quark-gluon plasma evolution. We now return to our motivating problem on the quark-gluon plasma, an exotic state of nuclear matter which can be created in modern particle colliders and which pervaded the universe during its first microseconds. The study of this plasma—in particular, properties of this unique phase of matter—is thus an important problem in high-energy nuclear physics. Modern investigation of QGP often requires computationally intensive numerical simulations, with the plasma modeled via relativistic fluid dynamics. The use of cost-efficient emulators, when carefully constructed, can thus greatly speed up the discovery of fundamental properties on the QGP, as evidenced in recent works [5].

For this study, we adopt a simplified version of the QGP simulation framework in [15], which can be split into three distinct stages: a pre-hydrodynamic stage, a hydrodynamic stage, and a post-hydrodynamic stage. Figure 1 visualizes this conglomerate (multi-stage) simulation framework. Each stage typically involves the discretization of the simulated physical system onto a spatial or space-time mesh (see Figure 8). The sizes and dimensionalities of the meshes may vary among stages. Meshes must be large enough to contain the entire initial and final states of the systems, to be fine enough to capture relevant details (e.g., small-scale fluctuations in the pre-hydrodynamic initial state), and yet to allow for timely computation.

The considered simulator has two key fidelity parameters, t_η and t_τ, which control its precision. The first fidelity parameter arises in the pre-hydrodynamic stage. This stage models the initial distribution of energy resulting from the collision of two atomic nuclei. The energy distribution is defined on a 3D (spatial) mesh with coordinates x, y, and η. The bounds of the mesh are fixed in all three dimensions, but the mesh density in the η-direction will be varied to adjust fidelity; it is specified by the longitudinal mesh size variable t_η, which serves as our first fidelity parameter. The simulation costs of all three stages are inversely proportional to t_η. The second parameter arises when the initial hydrodynamic state is evolved with the relativistic hydrodynamic equations until a completion criterion is reached, in effect extending the mesh into a time dimension, denoted by τ. The temporal mesh size variable t_τ—our second fidelity parameter—can thus be varied to adjust fidelity, although we note that in contrast with the η spatial direction, the number of time-steps is not known in advance

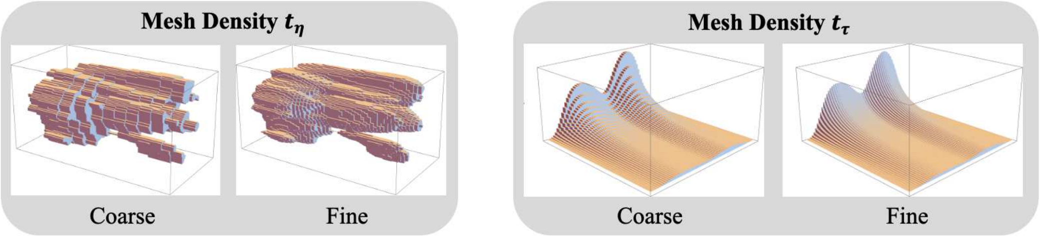


Figure 8. *Visualizing the two mesh densities (fidelity parameters) in the quark-gluon plasma simulation.*

because the full evolution time cannot be fixed: it is only determined once the completion criterion is satisfied and hence depends on the initial conditions in a complicated way. Except at very low fidelity, the simulation costs of the hydrodynamic and post-hydrodynamic stages are inversely proportional to t_τ.

In this simplified QGP simulator, we consider a single response variable: the ratio of pions produced at two different points, $\eta = 0$ and $\eta = 1$. This ratio serves as a measure of how particle production is distributed along the collision axis of the atomic nuclei and is chosen because it is strongly influenced by the model parameter α, which we use as our single input parameter in this study. We denote α as x_1 and the ratio observable as y_1 below.

The experiment is carried out as follows. We compare the CONFIG models with the standard GP and the TWY models. Since the multi-stage procedure involves multiple sequential stages, it falls under Scenario 2 (see section 3), and thus we recommend the use of Kernel 2 for CONFIG, although results for Kernel 1 are included later for completeness. We set $\ell = \ell_r = 2$ for CONFIG and $\ell = 2$ for the two TWY models; these were optimized via cross-validation. To demonstrate the cost efficiency of multi-fidelity emulation, we again include the "high-fidelity GP" model, which makes use of *only* high-fidelity runs to train a standard GP emulator using the squared-exponential kernel. As before, the limiting highest-fidelity setting of $\mathbf{t} = \mathbf{0}$ cannot be numerically simulated. We thus set the fidelity parameters $\mathbf{t} = (1.0 \times 10^{-4}, 1/64)$ as the "high-fidelity" setting for prediction, which appears to provide a fine enough mesh according to a mesh validation study. With this, a single high-fidelity run is very time-consuming, requiring around 1,000 CPU hours.

For comprehensive cost analysis, we fit each emulator using different design sizes and then compare the predictive performance of these models given a computational budget. The training data are generated as follows. For the high-fidelity GP model, we generate $n = 2, 3, 4,$ or 5 maximin (equally spaced) high-fidelity design points over the input interval $x_1 \in [3, 5]$. For the remaining models, we generate $n = 15$, 20, or 25 design points over the joint space of input and fidelity parameters. Each design has an equal number of points on five maximin (equally spaced) levels on x_1. For the two fidelity parameters t_τ and t_η, we first generate a 2D n-point maximin LHD [42] and scale this over the domain $[1.0 \times 10^{-4}, 5.0 \times 10^{-2}] \times [1/64, 1/24]$. We then randomly assign to each level of x_1 a fidelity setting from this LHD. For validation, the test set is generated on 100 evenly spaced points over the input space $x_1 \in [3, 5]$, run at the aforementioned high-fidelity setting for $\mathbf{t}$. To account for simulation variation, we repeat this procedure of training data generation and model fitting 20 times and compute the average of all metrics. The average computational cost for generating multi-fidelity training data ranges from 2.4×10^3 to 3.7×10^3 CPU hours for 15 to 25 design points, respectively.

Consider first the comparison of the predictive performance of the emulators given a computational budget for training data generation. Figure 9(left) plots the log-MSEs of the considered models and their corresponding costs (in log10-CPU hours) for simulating the training data. We see that, at a given computational budget, CONFIG with Kernel 2 (as recommended under Scenario 2) yields the best predictive performance out of all methods. This suggests that by integrating information on the underlying conglomerate (multi-stage) multi-fidelity simulation framework within its kernel specification, the proposed model can provide *cost-efficient* and accurate emulation of expensive simulators given a tight computational budget. While such errors appear relatively small in magnitude, it is shown in [37]

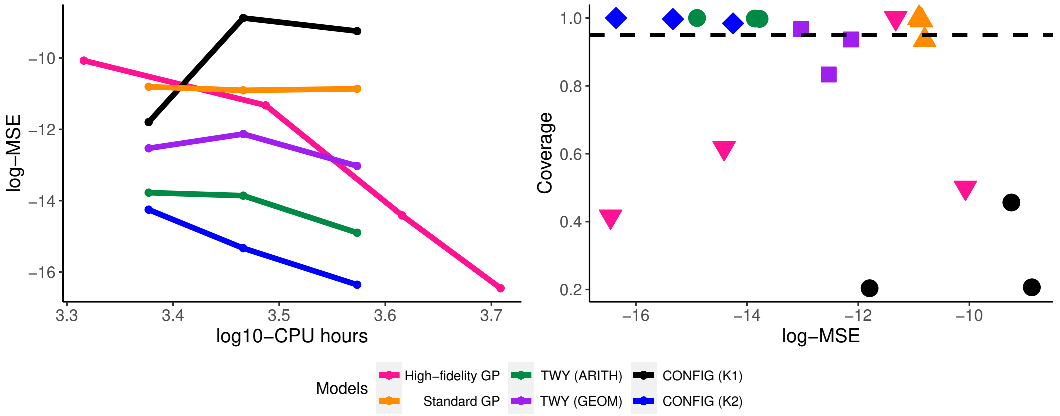


Figure 9. *(Left) Plot of testing log-MSE vs.* log10*-CPU hours required for training data simulation for each emulator model. (Right) Scatterplot of empirical coverage ratios vs. log-MSEs for the compared models. The black dashed line denotes the nominal* 95% *rate.*

that small improvements in emulation accuracy may lead to large improvements (i.e., tighter constraints) for Bayesian parameter estimation of QGP properties. As such, the improved predictions from CONFIG can facilitate greater precision in scientific studies. It is also interesting to note the poor performance of CONFIG with Kernel 1 here, which we do not recommend using since this falls under Scenario 2. This is not too surprising given the fewer interactions between fidelity parameters in the current sequential multi-stage setting.

For coverage ratios, Figure 9(right) shows the scatterplot of empirical coverage ratios (for 95% predictive intervals) vs. log-MSEs for the compared methods. For the high-fidelity GP, we see that while it can achieve relatively low MSEs, it has severe undercoverage and the required cost for training data generation is high. The standard GP and the TWY models, on the other hand, provide good coverage but poor predictive performance. Of the compared models, CONFIG with the recommended Kernel 2 yields the best predictive performance with good coverage. This again suggests that, by integrating the underlying conglomerate (multi-stage) multi-fidelity framework for non-stationary kernel specification, CONFIG can yield improved emulation performance with reliable uncertainty quantification.

7. Conclusion.

In this paper, we presented a new emulator model, called the CONFIG model, that tackles the challenge of surrogate modeling for conglomerate multi-fidelity simulators, whose precision is controlled by multiple fidelity parameters. Such simulators are often encountered in complex physical systems (including our motivating application in high-energy nuclear physics), but there has been little work in constructing cost-efficient emulators which leverage this structure for predictive modeling. CONFIG makes use of novel non-stationary covariance functions, which embed numerical convergence information on the underlying conglomerate simulator within its kernel specification. This infusion of prior information allows for effective surrogate modeling of complex simulators, even with limited training data. We demonstrate the effectiveness of the CONFIG model in a suite of simulation experiments and two applications, the first on emulating cantilever beam deflection and the second on emulating the quark-gluon plasma in high-energy physics.

With these encouraging results, there are many avenues for future work. Given the promise of multi-fidelity modeling, one crucial direction for maximizing predictive power given a tight computational budget is experimental design. While there is a growing literature on design for multi-fidelity modeling [16, 21, 36, 61, 71], such methods largely do not account for multiple fidelity parameters (as is present in conglomerate simulators) or factor in varying simulation costs. For example, given a budget of 10^6 CPU hours for a project, an experimenter would wish to know if a better predictive model can be trained with a few carefully chosen higher-fidelity runs or with more lower-fidelity runs. Tackling this design problem for the current conglomerate multi-fidelity framework can greatly increase the applicability of CONFIG in applications. We also aim to extend the CONFIG model for a broader range of multi-fidelity applications, where simulator fidelity is more complex and cannot be well-captured by several continuous fidelity parameters.

Acknowledgments. We greatly appreciate comments and suggestions from the anonymous referees, which have improved the quality of this paper. We further thank the JETSCAPE collaboration (https://jetscape.org/) for stimulating discussions and conversations that directly motivated this work.

REFERENCES

[1] J. Bect, S. Zio, G. Perrin, C. Cannamela, and E. Vazquez, *On the quantification of discretization uncertainty: Comparison of two paradigms*, in 14th World Congress in Computational Mechanics and ECCOMAS Congress 2020 (WCCM-ECCOMAS), 2021.

[2] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, 3rd ed., Springer, New York, 2008.

[3] S. P. Brooks and A. Gelman, *General methods for monitoring convergence of iterative simulations*, J. Comput. Graph. Stat., 7 (1998), pp. 434–455.

[4] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, *A limited memory algorithm for bound constrained optimization*, SIAM J. Sci. Comput., 16 (1995), pp. 1190–1208, https://doi.org/10.1137/0916069.

[5] S. Cao, Y. Chen, J. Coleman, J. Mulligan, P. Jacobs, R. Soltz, A. Angerami, R. Arora, S. Bass, L. Cunqueiro, et al., *Determining the jet transport coefficient $\hat{q}$ from inclusive hadron suppression measurements using Bayesian parameter estimation*, Phys. Rev. C, 104 (2021), 024905.

[6] A. Chakraborty, S. Gopalakrishnan, and J. N. Reddy, *A new beam finite element for the analysis of functionally graded materials*, Int. J. Mech. Sci., 45 (2003), pp. 519–539.

[7] J. Chen, S. Mak, V. R. Joseph, and C. Zhang, *Function-on-function kriging, with applications to three-dimensional printing of aortic tissues*, Technometrics, 63 (2021), pp. 384–395.

[8] C. Chivers, *MHadaptive: General Markov Chain Monte Carlo for Bayesian Inference Using Adaptive Metropolis-Hastings Sampling*, R package version 1.1-8, 2012, https://cran.r-project.org/package=MHadaptive.

[9] D. D. Cox, J.-S. Park, and C. E. Singer, *A statistical method for tuning a computer code to a data base*, Comput. Stat. Data Anal., 37 (2001), pp. 77–92.

[10] W. C. Cui and M. R. Wisnom, *Contact finite element analysis of three-and four-point short-beam bending of unidirectional composites*, Compos. Sci. Technol., 45 (1992), pp. 323–334.

[11] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker, *Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments*, J. Amer. Statist. Assoc., 86 (1991), pp. 953–963.

[12] R. D. De Souza, T. Koide, and T. Kodama, *Hydrodynamic approaches in relativistic heavy ion reactions*, Prog. Part. Nucl. Phys., 86 (2016), pp. 35–85.

[13] F. A. DiazDelaO and S. Adhikari, *Bayesian assimilation of multi-fidelity finite element models*, Comput. Struct., 92 (2012), pp. 206–215.

[14] R. Durrett, *Probability: Theory and Examples*, Camb. Ser. Stat. Probab. Math. 49, Cambridge University Press, Cambridge, UK, 2019.

[15] D. Everett, W. Ke, J. F. Paquet, G. Vujanovic, S. A. Bass, L. Du, C. Gale, M. Heffernan, U. Heinz, D. Liyanage, et al., *Multisystem Bayesian constraints on the transport coefficients of QCD matter*, Phys. Rev. C, 103 (2021), 054904.

[16] A. I. Forrester, A. Sóbester, and A. J. Keane, *Multi-fidelity optimization via surrogate modelling*, Proc. A, 463 (2007), pp. 3251–3269.

[17] T. E. Fricker, J. E. Oakley, and N. M. Urban, *Multivariate Gaussian process emulators with nonseparable covariance structures*, Technometrics, 55 (2013), pp. 47–56.

[18] C. Gale, S. Jeon, and B. Schenke, *Hydrodynamic modeling of heavy-ion collisions*, Int. J. Modern Phys. A, 28 (2013), 1340011.

[19] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, Chapman and Hall/CRC, Boca Raton, FL, 1995.

[20] A. Gelman and D. B. Rubin, *Inference from iterative simulation using multiple sequences*, Statist. Sci., 7 (1992), pp. 457–472.

[21] S. F. Ghoreishi and D. Allaire, *Multi-information source constrained Bayesian optimization*, Struct. Multidiscip. Optim., 59 (2019), pp. 977–991.

[22] E. Gul, V. R. Joseph, H. Yan, and S. N. Melkote, *Uncertainty quantification of machining simulations using an in situ emulator*, J. Qual. Technol., 50 (2018), pp. 253–261.

[23] X. He, R. Tuo, and C. F. J. Wu, *Optimization of multi-fidelity computer experiments via the EQIE criterion*, Technometrics, 59 (2017), pp. 58–68.

[24] U. Heinz and R. Snellings, *Collective flow and viscosity in relativistic heavy-ion collisions*, Annu. Rev. Nucl. Partial Sci., 63 (2013), pp. 123–151.

[25] P. R. Heyliger and J. N. Reddy, *A higher order beam finite element for bending and vibration problems*, J. Sound Vib., 126 (1988), pp. 309–326.

[26] M.-F. Ho, S. Bird, and C. R. Shelton, *Multifidelity emulation for the matter power spectrum using Gaussian processes*, Mon. Not. Roy. Astron. Soc., 509 (2022), pp. 2551–2565.

[27] W. H. Hundsdorfer, J. G. Verwer, and W. Hundsdorfer, *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, Springer Ser. Comput. Math. 33, Springer-Verlag, Berlin, 2003.

[28] Y. Ji, S. Mak, D. Soeder, J.-F. Paquet, and S. A. Bass, *A graphical multi-fidelity Gaussian process model, with application to emulation of heavy-ion collisions*, Technometrics, 66 (2024), pp. 267–281.

[29] S. S. Jin, S. T. Kim, and Y. H. Park, *Combining point and distributed strain sensor for complementary data-fusion: A multi-fidelity approach*, Mech. Syst. Signal Process., 157 (2021), 107725.

[30] M. E. Johnson, L. M. Moore, and D. Ylvisaker, *Minimax and maximin distance designs*, J. Stat. Plann. Inference, 26 (1990), pp. 131–148.

[31] V. R. Joseph, *Space-filling designs for computer experiments: A review*, Qual. Eng., 28 (2016), pp. 28–35.

[32] V. R. Joseph, E. Gul, and S. Ba, *Maximum projection designs for computer experiments*, Biometrika, 102 (2015), pp. 371–380.

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

[34] J. Kou and W. Zhang, *Multi-fidelity modeling framework for nonlinear unsteady aerodynamics of airfoils*, Appl. Math. Model., 76 (2019), pp. 832–855.

[35] A. Kumar, Y. Tachibana, C. Sirimanna, G. Vujanovic, S. Cao, A. Majumder, Y. Chen, L. Du, R. Ehlers, D. Everett, et al., *Inclusive jet and hadron suppression in a multistage approach*, Phys. Rev. C, 107 (2023), 034911.

[36] L. Le Gratiet and C. Cannamela, *Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes*, Technometrics, 57 (2015), pp. 418–427.

[37] D. Liyanage, Y. Ji, D. Everett, M. Heffernan, U. Heinz, S. Mak, and J.-F. Paquet, *Efficient emulation of relativistic heavy ion collisions with transfer learning*, Phys. Rev. C, 105 (2022), 034910, https://doi.org/10.1103/PhysRevC.105.034910.

[38] S. Mak and V. R. Joseph, *Minimax and minimax projection designs using clustering*, J. Comput. Graph. Statist., 27 (2018), pp. 166–178.

[39] S. Mak, C. L. Sung, X. Wang, S. T. Yeh, Y. H. Chang, V. R. Joseph, V. Yang, and C. F. J. Wu, *An efficient surrogate model for emulation and physics extraction of large eddy simulations*, J. Amer. Statist. Assoc., 113 (2018), pp. 1443–1456.

[40] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *Equation of state calculations by fast computing machines*, J. Chem. Phys., 21 (1953), pp. 1087–1092.

[41] S. T. More and R. S. Bindu, *Effect of mesh size on finite element analysis of plate structure*, Int. J. Eng. Sci. Innov. Technol., 4 (2015), pp. 181–185.

[42] M. D. Morris and T. J. Mitchell, *Exploratory designs for computational experiments*, J. Statist. Plann. Inference, 43 (1995), pp. 381–402.

[43] H. B. Moss, D. S. Leslie, J. Gonzalez, and P. Rayson, *Gibbon: General-purpose information-based Bayesian optimisation*, J. Mach. Learn. Res., 22 (2021), pp. 10616–10664.

[44] S. R. Narayanan, Y. Ji, H. D. Sapra, S. Yang, S. Mak, Z. Sun, S. Kokjohn, K. Kim, and C. B. Kweon, *Physics-integrated segmented Gaussian process (SegGP) learning for cost-efficient training of diesel engine control system with low cetane numbers*, in AIAA SCITECH 2023 Forum, 2023, 1283.

[45] D. Ngo and A. C. Scordelis, *Finite element analysis of reinforced concrete beams*, J. Am. Concr. Inst., 64 (1967), pp. 152–163.

[46] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.

[47] S. J. Park, Y. Y. Earmme, and J. H. Song, *Determination of the most appropriate mesh size for a 2-d finite element analysis of fatigue crack closure behaviour*, Fatigue Fract. Eng. Mater. Struct., 20 (1997), pp. 533–545.

[48] P. Perdikaris, M. Raissi, A. Damianou, N. D. Lawrence, and G. E. Karniadakis, *Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling*, Proc. A, 473 (2017), 20160751.

[49] P. Perdikaris, D. Venturi, J. O. Royset, and G. E. Karniadakis, *Multi-fidelity modelling via recursive co-kriging and Gaussian–Markov random fields*, Proc. A, 471 (2015), 20150018.

[50] V. Picheny and D. Ginsbourger, *A nonstationary space-time Gaussian process model for partially converged simulations*, SIAM/ASA J. Uncertain. Quantif., 1 (2013), pp. 57–78, https://doi.org/10.1137/120882834.

[51] M. Plummer, N. Best, K. Cowles, and K. Vines, *CODA: Convergence diagnosis and output analysis for MCMC*, R News, 6 (2006), pp. 7–11, https://journal.r-project.org/articles/RN-2006-002/RN-2006-002.pdf.

[52] M. Poloczek, J. Wang, and P. Frazier, *Multi-information source optimization*, in Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017, pp. 4291–4301.

[53] P. Z. Qian and C. F. J. Wu, *Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments*, Technometrics, 50 (2008), pp. 192–204.

[54] Z. Qian, C. C. Seepersad, V. R. Joseph, J. K. Allen, and C. F. J. Wu, *Building surrogate models based on detailed and approximate simulations*, J. Mech. Design, 128 (2006), pp. 668–677.

[55] C. S. Reese, A. G. Wilson, M. Hamada, H. F. Martz, and K. J. Ryan, *Integrated analysis of computer and physical experiments*, Technometrics, 46 (2004), pp. 153–164.

[56] P. Roache, *Perspective: A method for uniform reporting of grid refinement studies*, J. Fluids Eng., 116 (1994), pp. 405–413.

[57] T. J. Santner, B. J. Williams, W. I. Notz, and B. J. Williams, *The Design and Analysis of Computer Experiments*, 1, Springer-Verlag, New York, 2003.

[58] R. Shi, L. Liu, T. Long, Y. Wu, and G. G. Wang, *Multi-fidelity modeling and adaptive co-kriging-based optimization for all-electric geostationary orbit satellite systems*, J. Mech. Des., 142 (2020), 021404.

[59] M. Smith, *ABAQUS/Standard User's Manual, Version* 6.9, Dassault Systèmes Simulia Corp., Providence, RI, 2009.

[60] A. Stein and L. Corsten, *Universal kriging and cokriging as a regression procedure*, Biometrics, 47 (1991), pp. 575–587.

[61] R. Stroh, J. Bect, S. Demeyer, N. Fischer, D. Marquis, and E. Vazquez, *Sequential design of multi-fidelity computer experiments: Maximizing the rate of stepwise uncertainty reduction*, Technometrics, 64 (2022), pp. 199–209.

[62] F. Sun, R. B. Gramacy, B. Haaland, S. Lu, and Y. Hwang, *Synthesizing simulation and field data of solar irradiance*, Stat. Anal. Data Min., 12 (2019), pp. 311–324.

[63] G. Sun, G. Li, M. Stone, and Q. Li, *A two-stage multi-fidelity optimization procedure for honeycomb-type cellular materials*, Comput. Mater. Sci., 49 (2010), pp. 500–511.

[64] J. A. Templeton, M. L. Blaylock, S. P. Domino, J. C. Hewson, P. R. Kumar, J. Ling, H. N. Najm, A. Ruiz, C. Safta, K. Sargsyan, A. Stewart, and G. Wagner, *Calibration and Forward Uncertainty Propagation for Large-Eddy Simulations of Engineering Flows*, Technical report, Sandia National Laboratory, Livermore, CA, 2015.

[65] P. Tighineanu, K. Skubch, P. Baireuther, A. Reiss, F. Berkenkamp, and J. Vinogradska, *Transfer learning with Gaussian processes for Bayesian optimization*, in Proceedings of the International Conference on Artificial Intelligence and Statistics (PMLR 2022), 2022, pp. 6152–6181.

[66] R. Tuo, C. F. J. Wu, and D. Yu, *Surrogate modeling of computer experiments with different mesh densities*, Technometrics, 56 (2014), pp. 372–380.

[67] E. Vanden-Eijnden, *Numerical techniques for multi-scale dynamical systems with stochastic effects*, Commun. Math. Sci., 1 (2003), pp. 385–391.

[68] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, Vol. 2, MIT Press, Cambridge, MA, 2006.

[69] C. F. J. Wu and M. S. Hamada, *Experiments: Planning, Analysis, and Optimization*, John Wiley & Sons, New York, 2009.

[70] S. Xiong, P. Z. Qian, and C. F. J. Wu, *Sequential design and analysis of high-accuracy and low-accuracy computer codes*, Technometrics, 55 (2013), pp. 37–46.

[71] H. Yuchi, V. R. Joseph, and C. F. J. Wu, *Design and analysis of multifidelity finite element simulations*, J. Mech. Des., 145 (2023), 061703.