Multi-fidelity Surrogate Modeling for Application/Architecture Co-design

Yiming Zhang[1], Aravind Neelakantan[2], Nalini Kumar[2], Chanyoung Park[1], Raphael T. Haftka[1], Nam H. Kim[1], and Herman Lam[2]

[1] Department of Mechanical and Aerospace Engineering
{yimingzhang521, cy.park, haftka, nkim}@ufl.edu
[2] Department of Electrical and Computer Engineering
{aravindneela,nkumar,hlam}@ufl.edu
University of Florida, Gainesville, Florida, 32608

Abstract. The HPC community has been using abstract, representative applications and architecture models to enable faster co-design cycles. While developers often qualitatively verify the correlation of the application abstractions to the parent application, it is equally important to quantify this correlation to understand how the co-design results translate to the parent application. In this paper, we propose a multi-fidelity surrogate (MFS) approach which combines data samples of low-fidelity (LF) models (representative apps and architecture simulation) with a few samples of a high-fidelity (HF) model (parent app). The application of MFS is demonstrated using a multi-physics simulation application and its proxy-app, skeleton-app, and simulation models. Our results show that RMSE between predictions of MFS and the baseline HF models was 4%, which is significantly better than using either LF or HF data alone, demonstrating that MFS is a promising approach for predicting the parent application performance while staying within a computational budget.

Keywords: performance estimation, multi-fidelity surrogate, behavioral emulation

1 Introduction

As we approach exascale computing, the next frontier in high-performance computing, it is important that application developers and system designers co-design to develop better performing and more energy efficient application codes and machines [6]. For fast and effective turnaround during the co-design process, application developers create representative applications which are abstract, smaller, and self-contained descriptions of their application code (also called parent app) and only capture the key parameters and features that predominantly influence the outcome of co-design [13]. To further speed up the co-design process and to enable architecture design-space exploration (DSE), system architects build simulator models to study the application performance on various underlying architectures. Behavioral Emulation (BE) [19] is one such coarse-grained approach for simulation of extreme-scale systems and applications. While the parent application can be used to drive architecture simulations, abstract application end-point models are often used to represent the parent app to speed up the co-design process.

Representative applications, in the form of mini-apps, proxy-apps, or skeleton apps, have been developed for many scientific HPC codes (parent app) and are a necessity in cases where the actual application cannot be shared with the hardware architects [5, 7, 16, 18]. After development, it is important to validate these representative apps against their parent apps to ensure they are reasonably accurate representations of the application behavior. Typically, this qualitative validation is performed by comparing ratio of computation to communication, weak scaling and strong scaling trends, similarity analysis, etc. Similarly, performance prediction results of architecture simulations are verified against testbed measurements. After validation, both the representative apps and simulator models can be used as platforms to evaluate tradeoffs for improved performance, power, and resilience, different programming models, compilers, etc. and guide the refinement of parent application. *Qualitative* validation is important; however, it is also important to determine *quantitatively* how the improvements in a representative app or architecture translate to the parent app. To the best of our knowledge no solution for quantitative validation of representative applications under a reasonable computational budget has been proposed in the literature.

In general, surrogate models are approximations that are fit to the available data of a phenomenon of interest, herein the parent app. A high-fidelity surrogate model (HFM) can be constructed from more accurate and computationally expensive high-fidelity data (e.g., benchmarking data using parent app); and a low-fidelity model (LFM) can be constructed from computationally cheaper but less accurate low-fidelity data (e.g., skeleton apps, simulation results). In this paper, we propose the use of a multi-fidelity surrogate model (MFS) for identifying the relation between parent and representative apps. The MFS works when LF and HF have similar trends/curvature in the design space. An indication of trend similarity between LF and HF is the scale factor. A scale factor around 1 denotes high correlation between HF and LF whereas a negative or extremely large scale factor denotes an inappropriate LF under certain discrepancy (discussed in Section 4).

The concept of MFS has been extensively studied to approximate the high-fidelity models (HFMs) assisted by cheaper low-fidelity models (LFMs) [12]. To balance accuracy and computational cost associated with data collection, the MFS approach aims to develop a surrogate model based mostly on LF samples assisted with only a few HF samples. Typical multi-fidelity models include finite element analysis with different resolution, physical tests versus numerical simulations, etc. [12, 17, 25].

Representative apps are often used for studying the performance impact of various optimization techniques. But it is important to validate these changes on the parent app. In order to use our proposed approach, changes have to be made to both the representative and parent app. The payoff of the additional effort required for modifying the parent app is the ability to validate performance over a considerably larger design space at a very low cost. In addition to low-cost validation of parent app, our proposed approach can also be used for predicting performance of the parent app with BE simulations of notional architectures as the source of LF data for developing the MFS. The HF data for notional architectures could be obtained from fine-grained simulators over a small subset of design-space. BE's ability to simulate any hardware through an architecture model, whether existing or notional, adds an additional capability of predicting performance of the parent app on the future systems quantitatively at a low cost.

In this paper, we leverage many of the MFS methods developed in other scientific domains and adapt and apply them to reduce the computational cost of validation of representative apps used in the HPC co-design process. After a survey of the related research in Section 2, in section 3 we present an overview of the parent application case study (CMT-nek), its representative mini-app (CMT-bone) and skeleton app (CMT-bone-BE), and the Behavioral Emulation approach that we use for performance modeling and simulation. In Sections 4 and 5, we describe a methodology for developing an MFS for an application from its mini-app (HFM), skeleton app (LFM), and a simulator model (LFM). In Section 6, we demonstrate the usefulness of the proposed methodology by applying it to a multi-physics simulation application being developed for exascale systems - CMT-nek [1]. The results demonstrate MFS as a promising approach for predicting parent application performance (HF model) from representative app or architecture simulation (LF model) while staying within a reasonable computational budget.

2 Related Research

Mini-apps have become extremely important for exascale DSE and performance optimization. In [9], which presents a validation methodology, the authors state that mini-apps reduce the DSE time by a factor of a thousand, making them extremely useful for exploring the design space of the parent application. Heroux, et al. in [10] provide a verification and validation (V&V) methodology for assessing the ability of the mini-app to effectively represent the performance of their parent application. The authors use the difference between mini-app and parent app performance as their validation metric and compare it against a threshold. This approach requires equal number of samples for both the parent app and the mini-app. Since samples for the parent app are typically more expensive to obtain, it can be a limiting factor in extensive validation studies over a large design space. In our proposed approach, constructing an MFS requires much fewer samples of parent app than of the mini-app, thus considerably reducing the computational budget of conducting performance validation.

Mini-apps are used as a tool to evaluate optimization methods to improve the performance of the parent application. But improving the performance of the mini-app does not guarantee the same for the parent app, making it important to know how representative these mini-apps are of their parent app [13]. For example, in [13] the authors seek to improve the performance of the application on new and future systems using mini-apps. Although the optimizations applied improve the mini-app performance, the impact on actual application performance is not clear. In our work, an MFS for the parent application, built using high-fidelity application performance samples and lower-fidelity mini-app performance samples, can help us draw a relationship between the performance behavior of the two applications.

Several frameworks have been proposed to realize multi-fidelity modeling in various science and engineering domains [12,17,25]. In [17], a Bayesian framework has been applied to predict the data of nuclear radiation based on simulations. A variable fidelity optimization framework has been demonstrated for the design of engine piston [23]. In [25], a deterministic framework has been proposed to predict the strength of composite laminate based on finite element simulations. Large number of application of MFS to mechanical systems can be found in [12], which reports that the MFS reduces computational cost drastically while enabling desirable prediction accuracy. The MFS has also been adopted as a powerful tool for uncertainty propagation [21].

Various MFS frameworks have been proposed for different engineering applications. For example, the Bayesian MFS based on a scale factor has been applied to design buildings [11] and flapping flight [26]. The Bayesian MFS incorporating discrepancy function has been proposed [17, 22] as a popular MFS framework for various applications. This Bayesian framework is equivalent to the co-Kriging surrogate [20] with no prior information. Balabanov et al. [4] used a sequential deterministic MFS based on the discrepancy function to combine finite element simulation with different resolutions. Zhang et al. [24,25] proposed a simultaneous deterministic MFS based on the discrepancy function to combine experimental strength and finite element simulation for composite laminate. We can apply this methodology in our HPC community to save computational cost of parent application.

Sampling schemes for multi-fidelity models have been studied correspondingly. HF samples are usually a subset of LF samples. One representative all-at-once sampling strategy is the nested design sampling [15]. First, LF samples are generated using Latin Hypercube Sampling (LHS). Then the HF samples are generated by maximizing the minimum distance between all existing LF samples. Huang et al. [14] proposed a sequential sampling scheme for design optimization using Bayesian MFS. Either LF or HF samples are generated iteratively for design optimization. In our approach, we used Full Factorial Design (FFD) [8] for sampling as it is convenient for parametric study.

In this paper, we leverage many of the MFS methods developed in other scientific domains and adapt and apply them to the HPC domain to reduce the computational cost of validation of representative apps used in the co-design process.

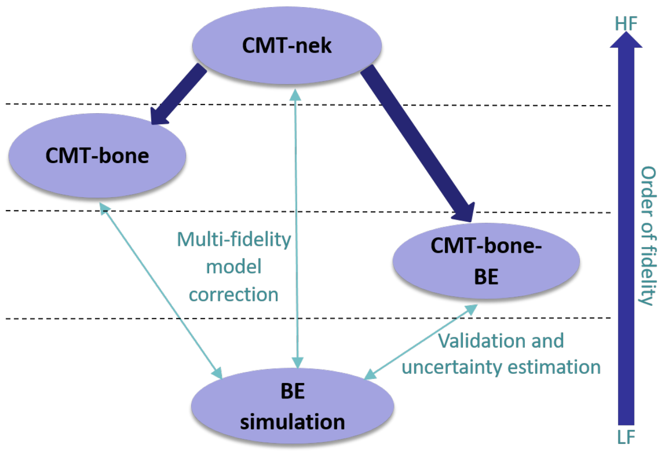

CMT-nek
CMT-bone
CMT-bone-BE
BE simulation
Multi-fidelity model correction
Validation and uncertainty estimation
HF
LF
Order of fidelity

Fig. 1. Hierarchy of the CMT models

3 Application and Architecture Models

In this section, we give an overview of the parent application under study (CMT-nek), its representative mini-app (CMT-bone) and skeleton app (CMT-bone-BE); and a BE simulation approach that we use for performance modeling and simulation. The relationships among their corresponding models are shown in Fig. 1. The parent app, CMT-nek, represents a high-fidelity (HF) model, whereas CMT-bone (a mini-app) and CMT-bone-BE (a skeleton app) are low-fidelity (LF) models, as compared to CMT-nek. BE simulation is a modeling and simulation of CMT-bone-BE. Thus, BE simulation is an even lower fidelity model than CMT-bone-BE and CMT-bone.

In this study, our objective is to first perform validation and uncertainty estimation of the BE simulation results against test samples of CMT-bone-BE (details in Section 5). Then a multi-fidelity surrogate model (MFS) is developed using mostly samples from the low-fidelity BE simulation and a few high-fidelity CMT-nek samples. The MFS model is then used to predict CMT-nek results (Section 6, case study 1). This experiment is repeated between BE simulation (LF) with CMT-bone (now being a HF, as compared to BE simulation) in case study 2, followed by CMT-nek (HF) and CMT-bone (LF) for case study 3.

3.1 CMT-nek

CMT-nek [1] is being developed at the PSAAP-II Center for Compressible Multiphase Turbulence (CCMT) at University of Florida to perform simulation of instabilities, turbulence, and mixing in particulate-laden flows under conditions of extreme pressure and temperature [1]. CMT has applications in many environmental, industrial, and national defense and security areas. CMT-nek is being developed from a production release of petascale code Nek5000 [2], a Gordon Bell prize winning open-source software for simulating unsteady incompressible fluid flow with thermal and passive scalar transport. It is a highly scalable code with strong scaling to over a million MPI ranks on ALCF BG/Q Mira. CMT-nek aims to take advantage of this sustained performance by inheriting the MPI strategies used in Nek5000; and by hooking into the Nek5000 repository, leveraging any changes and optimizations made to Nek5000.

3.2 CMT-bone and CMT-bone-BE

CMT-bone is a mini-app that encapsulates the key data structures and compute and communication kernels of CMT-nek. While retaining the workflow of CMT-nek, CMT-bone simplifies the number of variables defined and allocated and also the number of computation and communication operations performed at each time step in the simulation. The authors in [5,18] have validated mini-app CMT-bone with its parent app CMT-nek and found that the key compute kernels are well represented by the proxy application.

CMT-bone-BE is a skeleton app of CMT-nek created to support rapid algorithmic design-space exploration. It models the computation that happens within every simulation timestep to calculate the partial derivative and exchange data between nearby spectral element meshes. CMT-bone-BE ignores the initial problem setup including mesh generation. Mesh generation operations can be abstracted and replaced with a computation model in BE.

3.3 Behavioral Emulation (BE) Simulation

Behavioral Emulation (BE) is a coarse-grained modeling and simulation approach that aims to provide timely, flexible, and scalable estimates of application performance on existing and future system architectures. In BE, the complexity of large-scale system simulation is handled by simultaneously dividing the simulation into different levels of system abstraction (e.g., device, node, rack, system) and abstracting the behavior of the components at each of these levels. The coarse-grained component models mimic or emulate the observed execution behavior of the component instead of its cycle-accurate operation. There are two basic types of BE models - application BE objects (AppBEOs) and architecture BE objects (ArchBEOs).

BE simulations are used to predict the execution time of CMT-bone-BE and have computational cost less than that of CMT-bone-BE (and much less than that of CMT-nek and CMT-bone). In our study, BE simulation results are used to produce the LF data points which are used to construct MFS models to predict the execution time of CMT-nek and CMT-bone, using very few data points from these two applications, thus reducing the overall computational budget of the DSE process.

4 Multi-fidelity Surrogates

In engineering applications, it is common to have multiple models with different fidelities for solving the same problem such as finite element simulations with different grid resolutions, numerical simulations, and physical experiments. A high-fidelity model (HFM) represents the physical phenomenon more accurately than the low-fidelity model (LFM) but it is often very expensive. An MFS based approach uses both high-fidelity and low-fidelity datasets to approximate the HFM in the design space. An effective MFS is expected to make accurate prediction with limited budget for sampling. Fernández-Godino et al. [12] reviewed recent developments of MFS especially on effectiveness of applying MFS to practical design. Peherstorfer et al. [21] summarized the technical details of MFS for inference and uncertainty propagation.

MFS translates LFM against a few HF samples using an algebraic function. Typical MFS frameworks include two major components: (1) a model to define the relation between LFM and HFM, and (2) the scheme to find parameters of the MFS and associated uncertainty of prediction. The LFM could be translated to HFM through (1) a constant scale factor, or (2) scaling up the LFM and adding to a discrepancy function. After determining the form of algebraic function, the MFS could be developed either using Bayesian inference through Gaussian process, or using a least-square regression minimizing error between fitted model and data.

In this work, we investigate the feasibility of MFS to quantify and mitigate the difference between high-fidelity parent applications (e.g., CMT-nek) and low-fidelity simulations (e.g., BE simulation) in the area of co-design of large-scale system. The least-squares MFS (LS-MFS) [24] was selected for this feasibility study while balancing complexity and predictive capability.

$$\hat{f}_H(\boldsymbol{x}) = \rho \hat{f}_L(\boldsymbol{x}) + \hat{\delta}(\boldsymbol{x}) \tag{1}$$

The LS-MFS is built with two surrogates, $\hat{f}_L(\boldsymbol{x})$, a polynomial response surface (PRS) fitted to low-fidelity data, and $\hat{\delta}(\boldsymbol{x})$, the fitted discrepancy data (equation 1). The scale factor ρ and discrepancy function $\hat{\delta}(x)$ are obtained to minimize prediction error at the high-fidelity samples according to equations 2 and 3. $(\boldsymbol{x}_H, \boldsymbol{y}_H)$ denotes the high-fidelity dataset containing n samples.

$$\min_{\rho, \hat{\delta}(x)} : (\hat{\delta}(\boldsymbol{x}_H) - \boldsymbol{d}_H)^T (\hat{\delta}(\boldsymbol{x}_H) - \boldsymbol{d}_H) \tag{2}$$

$$\boldsymbol{d}_H = \rho \hat{f}_L(\boldsymbol{x}_H) - \boldsymbol{y}_H \tag{3}$$

The multi-fidelity surrogate using a single linear regression is obtained from equations (4-7). $\boldsymbol{Y}$ is the vector of high-fidelity samples, $\boldsymbol{X}$ is the augmented design matrix, $\boldsymbol{B}$ is the vector of unknown coefficients and $\boldsymbol{e}$ is the vector for residual errors. $\boldsymbol{X}_i(\boldsymbol{x})$ denotes the i^{th} monomial/basis, and b_i is the coefficient of $\boldsymbol{X}_i(\boldsymbol{x})$. The obtained discrepancy function $\hat{\delta}(\mathbf{x})$ is shown in equation 8.

$$\mathbf{Y} = \mathbf{XB} + \mathbf{e} \tag{4}$$

$$\mathbf{Y}_{n\times 1} = \begin{bmatrix} \mathbf{y}_H^{(1)} \\ \vdots \\ \mathbf{y}_H^{(n)} \end{bmatrix}, \mathbf{B}_{p+1} = \begin{bmatrix} \rho \\ b_1 \\ \vdots \\ b_p \end{bmatrix}, \mathbf{e}_{n\times 1} = \begin{bmatrix} \mathbf{e}_H^{(1)} \\ \vdots \\ \mathbf{e}_H^{(n)} \end{bmatrix} \tag{5}$$

$$\mathbf{X}_{n\times(p+1)} = \begin{bmatrix} \hat{f}_L(\mathbf{x}_H^{(1)}) & X_1(\mathbf{x}_H^{(1)}) & \dots & X_p(\mathbf{x}_H^{(1)}) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{f}_L(\mathbf{x}_H^{(n)}) & X_1(\mathbf{x}_H^{(n)}) & \dots & X_p(\mathbf{x}_H^{(n)}) \end{bmatrix} \tag{6}$$

$$\mathbf{B} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{7}$$

$$\hat{\delta}(\mathbf{x}) = \sum_{i=1}^{p} b_i X_i(\mathbf{x}) \tag{8}$$

The LS-MFS scales up the LFM and adds a polynomial function $\hat{\delta}(\boldsymbol{x})$ to match a few high-fidelity samples. The scale factor ρ is critical for prediction. Negative or extremely large values of ρ indicates a prediction with large error, which is likely to be associated with undesirable LFMs, inappropriate surrogate forms, or inadequate samples. $\hat{\delta}(\boldsymbol{x})$ is supposed to be a low-order polynomial function while assuming the the LFM has a trend similar to HFM. In our study, we adopted a constant $\hat{\delta}(\boldsymbol{x})$ for less than 10 HF samples and a linear polynomial function as $\hat{\delta}(\boldsymbol{x})$ for the rest. We approximated the execution time in logarithmic coordinate to account for the order-of-magnitude variation of execution time. $\hat{f}_L(\boldsymbol{x})$ was developed using a quartic PRS.

5 Developing MFS Model

Although various performance metrics can be studied for performance simulation such as energy consumption and communication times between the processors, the metric of interest in this paper is the total execution time for running a typical computational fluid dynamics analysis using CMT-nek (HFM), CMT-bone (HFM), CMT-bone BE (LFM), and BE simulation (LFM). All the benchmarking of the CMT models is performed on the Vulcan HPC platform from Lawrence Livermore National Laboratory (LLNL) [3]. Vulcan is a 24-rack IBM Blue Gene/Q system based on POWER architecture that consists of 24,576 nodes and 400TB of compute memory. It is important to ensure that the HF and LF data are obtained from the same hardware. The accuracy of the MFS model reflects how representative the LF and HF are of each other.

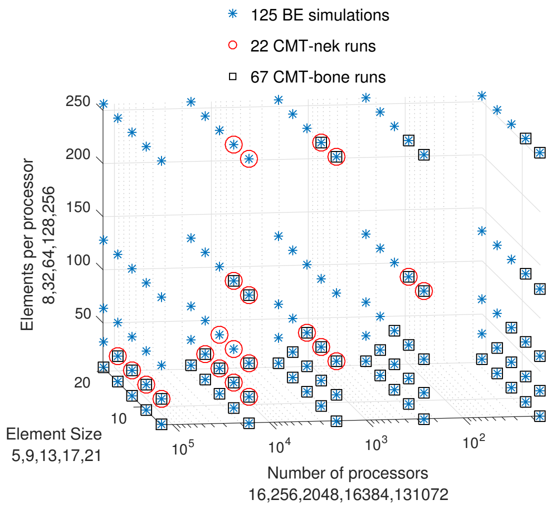


Fig. 2. Design of experiments for CMT-nek, CMT-bone, and BE simulations

5.1 Design Space

For CMT-nek, the three main application parameters of concern are Element Size (ES), Elements per Processor (EPP) and Number of Processors (NP). Application performance can be affected by changing any one of these parameters. We chose 125 experimental points based on five-level full factorial design. "Five-level" denotes the 5 points/grids selected along each application parameter with similar space, as shown in Fig. 2. The design of experiment is ES={5,9,13,17,21}, EPP={8,32,64,128,256} and NP={16,256,2048,16384,131072}. The experimental runs require up to 131,072 processors, 34 million elements and 311 billion computational grid points.

As the model fidelity increases so does the cost of obtaining a test sample. We obtained data for CMT-bone-BE and BE simulation for the entire design space (125 data points); but for CMT-nek and CMT-bone, data was judiciously obtained from a subset of the design space. For the runs from LFM and HFM, we made 22 runs from CMT-nek (HFM), 67 runs from CMT-bone, and 125 runs for both LFMs (CMT-bone-BE and BE simulation).

5.2 Validations of BE simulation results

Recall from Fig. 1 that the order of fidelity is as follows: parent app CMT-nek (highest), mini-app CMT-bone, skeleton app CMT-bone-BE, and BE simulation (lowest). In the next section, we will use the BE-simulation results (LF) to predict the performance of CMT-nek (HF parent app). Thus, first it is important to evaluate the accuracy of the BE simulation. To do so, in this section, we will first validate the accuracy of BE simulation against skeleton app CMT-bone-BE.

We then evaluate the accuracy of CMT-bone-BE by validating its results against those of mini-app CMT-bone.

BE simulation vs. CMT-bone-BE. Validation is the process of comparing the BE simulation results to its respective benchmarking result using CMT-bone-BE. In this study, we have validated the simulation results for the entire design space on Vulcan, one of the largest high-performance computing system available at the Lawrence Livermore National Lab. The validation of the design space covered all the calibration points. But to further evaluate the accuracy of the simulator, true validation was performed by validating points that are not present in the calibration set. Polynomial interpolation was used in the simulator to predict the execution time of the application at these validation points. The obtained simulation results are then validated by running the actual CMT-bone-BE application on Vulcan for those validation points.

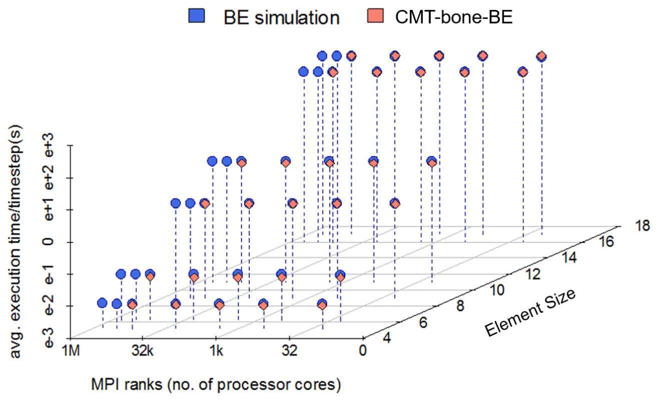
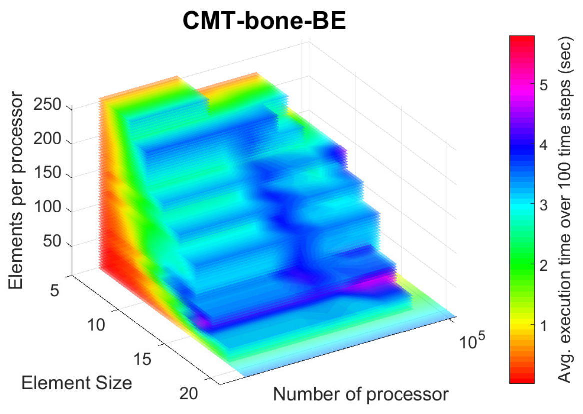


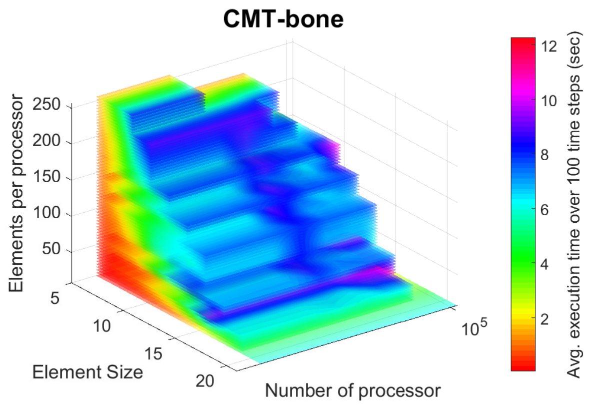
Fig. 3. Validation of BE simulation against CMT-bone-BE

The validation results of BE simulation against CMT-bone-BE are shown in Fig. 3 where the blue points represent the predicted CMT-bone-BE time using BE simulation and the red points are the validation points obtained by running the application (CMT-bone-BE) on Vulcan. We validated the simulations up to 128k NP on Vulcan and predicted the time for 256k NP and 512k NP. The average percentage error between BE simulation and CMT-bone-BE is 4%, thus demonstrating the accuracy of the BE simulator.

CMT-bone-BE vs. CMT-bone. The validation of the skeleton app CMT-bone-BE against mini-app CMT-bone (Fig. 4a and 4b, respectively) is done through comparing their trends under the same experimental setup described above. CMT-bone-BE being the skeleton app, takes less time to execute than that of the mini-app, CMT-bone, and hence the range of their execution time varies. Therefore, to make it easier to compare the trend, the execution time is plotted on a color scale with red being the lowest in the range and blue



(a)



(b)

Fig. 4. Comparing CMT-bone mini-app and CMT-bone-BE skeleton app trends for various parameter values

being highest in the range as shown in Fig. 4. The step-wise increase shows that the predicted execution time for CMT-bone-BE and CMT-bone increases monotonically with ES and EPP and does not change appreciably with NP, and the color scale on the graphs verify the similarity in trend between CMT-bone-BE and CMT-bone.

6 Evaluating MFS Predictions — Three Case Studies

Three case studies were used to demonstrate the multi-fidelity surrogate (MFS) approach in which a surrogate model, based mostly on low-fidelity (LF) samples assisted with only a few high-fidelity (HF) samples, is used to predict the performance of a high-fidelity (HF).

- Case 1: Multi-fidelity model based mostly on BE simulation (LF) and few CMT-nek (HF parent app) data points to predict the performance of CMT-nek (HF)
- Case 2: Multi-fidelity model based mostly on BE simulation (LF) and few CMT-bone (relatively HF mini-app) data points to predict the performance of CMT-bone (HF)
- Case 3: Multi-fidelity model based mostly on CMT-bone (relatively LF mini-app) and few CMT-nek (HF) data points to predict the performance of CMT-nek (HF)

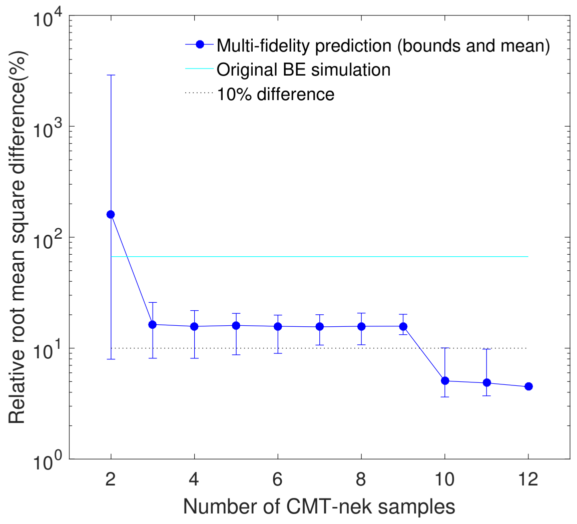
The setup was same in all three case studies. A subset of high-fidelity data was selected as the validation/test runs to evaluate predictions while the others were used as training runs to train LS-MFS (least square MFS). The number of samples increased gradually from the remaining runs (which excludes the validation runs) to investigate the effect of sampling plan. For each number of samples, random selection was repeated 20 times to account for the effect of sampling plan. The overall difference was measured using relative root-mean-square error (R-RMSE) between the LS-MFS predictions and the validation runs. The relative maximum difference (R-MD) at the validation runs based on the repeated samples was also provided to understand individual prediction. In this paper, we study LS-MFS using polynomial response surface as it is robust with noise effect. Other frameworks of multi-fidelity surrogates are also available such as co-Kriging. The comparison between different multi-fidelity surrogates is beyond the scope of this paper.

Table 1. Predicting execution times of CMT models based on typical set of 12 samples and evaluating the prediction using R-RMSE (%)

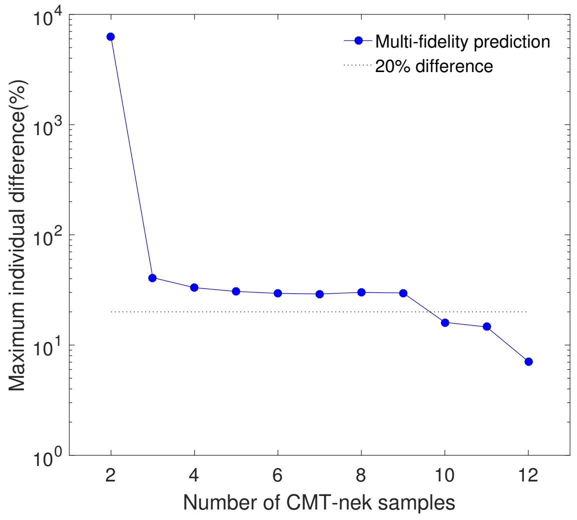
	Case 1: CMT-nek prediction from BE simulation	Case 2: CMT-bone prediction from BE simulation	Case 3: CMT-nek prediction from CMT-bone
Number of validation runs	10	20	10
LS-MFS	4.49%	5.40%	7.34%
$\hat{f}_L(\boldsymbol{x})$	66.85%	61.26%	18.65%
Linear fit to $(\boldsymbol{x}_H, \boldsymbol{y}_H)$	131.23%	1901.91%	131.23%
Residual errors of $\hat{f}_L(\boldsymbol{x})$	0.77%	0.77%	1.05%
Residual errors of the linear fit to $(\boldsymbol{x}_H, \boldsymbol{y}_H)$	18.40%	50.71%	18.40%

6.1 Case study 1: CMT-nek predictions from BE simulations

22 runs of CMT-nek are obtained as shown in Fig. 2. 10 runs (out of 22) were selected randomly and fixed as the validation runs. We first examined LS-MFS to approximate CMT-nek (HF model) runs using a typical set of 12 samples as shown in the Case 1 column of Table 1. The R-RMSE of LS-MFS is 4.49%



(a) R-RMSE



(b) R-MD

Fig. 5. Difference between CMT-nek validation runs and multi-fidelity predictions based on BE simulation

using BE simulation (LF model) at 10 validation runs. The linear fit using only HF CMT-nek runs was also developed as a comparison with the R-RMSE to be 131.23% at the validation runs. The R-RMSE between original BE simulation and CMT-nek ($\hat{f}_L(\boldsymbol{x})$) at all the 12 points, without translation, is 66.85%. As mentioned before, BE simulation mimics CMT-bone-BE and not CMT-nek. Since CMT-bone-BE is just a skeleton app with very few computational kernels, the percentage difference is high. The LS-MFS was much more accurate than either the $\hat{f}_L(\boldsymbol{x})$ or the linear fit to $(\boldsymbol{x}_H, \boldsymbol{y}_H)$, demonstrating its promise to compensate the difference between high-fidelity and low-fidelity models.

Next, we investigated the effect of the sampling plan on prediction accuracy. The LS-MFS predictions for CMT-nek runs with increasing number of samples were summarized in Fig. 5a. The LS-MFS was unstable using only 2 CMT-nek samples due to over-fitting and became more accurate with increasing CMT-nek samples. The R-RMSE was less than 10% with more than 9 CMT-nek samples and ended with 4.49%. The R-MD was less than 20% with more than 9 CMT-nek samples and ended with 7% as seen in Fig. 5b. The order of $\hat{\delta}(\boldsymbol{x})$ was changed from constant to linear for more than 9 CMT-nek samples which was critical for the accuracy of LS-MFS. We specified the order of $\hat{\delta}(\boldsymbol{x})$ for simplicity in this feasibility study. The performance of LS-MFS could be improved by choosing appropriate $\hat{\delta}(\boldsymbol{x})$. Another observation is the large variation of R-RMSE while repeating HF samples. The design of experiments for HF samples affected LS-MFS noticeably. The evaluation of LS-MFS for CMT-nek is based on up to 12 samples and may suffer large uncertainty due to the scarce runs.

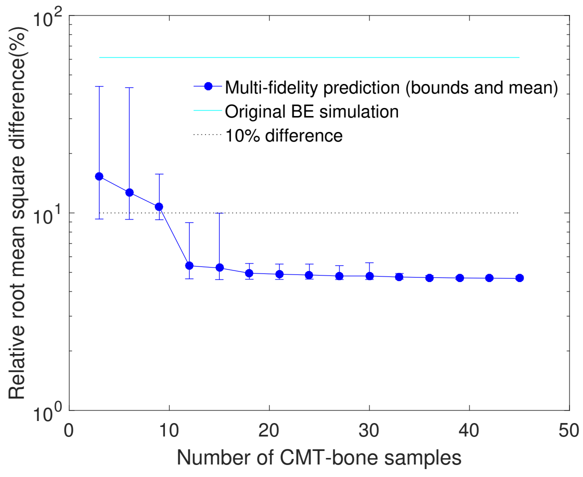
6.2 Case study 2: CMT-bone predictions from BE simulations

20 runs (out of 67) were selected randomly and fixed as the validation runs. We performed LS-MFS for CMT-bone (HF model in this case) based on 20 validation runs and up to 47 samples. Again, LS-MFS was most accurate comparing to $\hat{f}_L(\boldsymbol{x})$ and the linear fit to only $(\boldsymbol{x}_H, \boldsymbol{y}_H)$ as shown in Case 2 column of Table 1. The LS-MFS predictions were much closer to HFM than the original BE simulations.

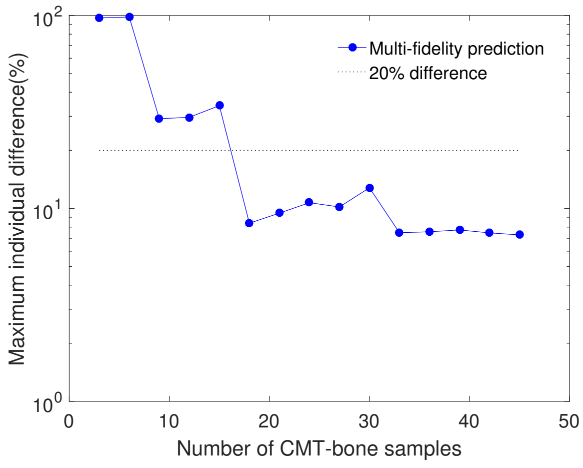
Once again, we investigate the effect of the sampling plan on prediction accuracy. The R-RMSE in Fig. 6a reduced with increasing CMT-bone samples and ended with 5.4%. The R-MD in Fig. 6b oscillated with scarce CMT-bone runs at the beginning and stabilized around 10%. Both R-RMSE and R-MD reduced noticeably with the first few samples and stabilized to less than 10% thus proving to be a promising approach.

6.3 Case study 3: CMT-nek predictions from CMT-bone

In the final case study, the LS-MFS was developed to predict the high-fidelity parent app (CMT-nek) from its low-fidelity mini-app (CMT-bone). This helps in quantitative validation of the mini-app. The setup was same as in case study 1, where CMT-nek was the HF model. From Case 3 column of Table 1, we see that LS-MFS provides the best fit compared to linear fit. The LS-MFS had less-than



(a) R-RMSE

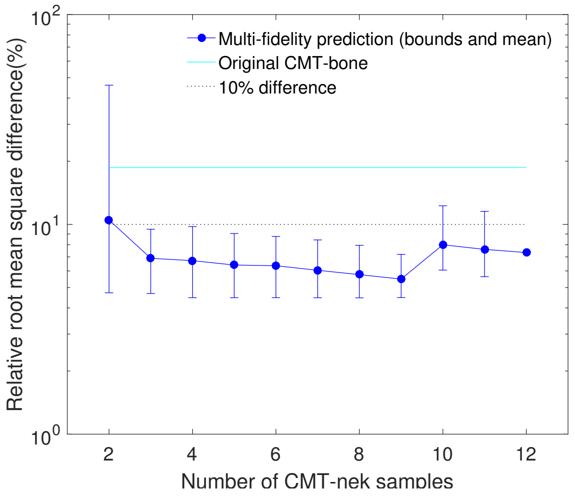


(b) R-MD

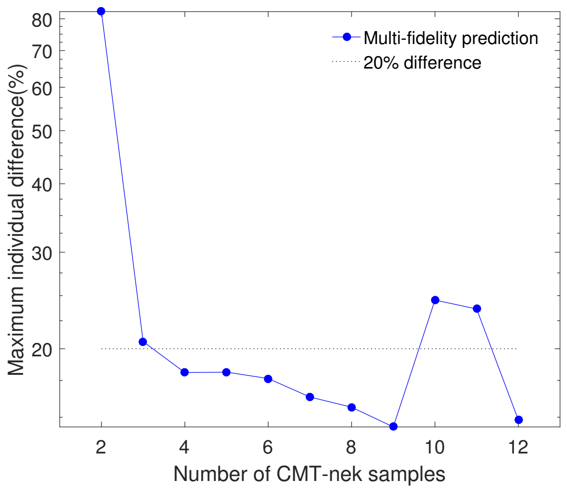
Fig. 6. Difference between CMT-bone validation runs and multi-fidelity predictions based on BE simulation

10% R-RMSE in Fig. 7a. It is worth noting the significant jump between 9 and 10 samples while changing the order of $\hat{\delta}(\boldsymbol{x})$ in Fig. 7a and 7b. CMT-bone was close to CMT-nek and a different scheme might be preferred to determine $\hat{\delta}(\boldsymbol{x})$.

A key observation between case study 1 and 3 is that although the CMT-bone samples were much closer to the CMT-nek samples, the MFS predictions of CMT-nek (HF) from BE simulations (LF) were more accurate than the MFS predictions from CMT-bone (LF) as shown in Table 1. Fitting CMT-bone was more challenging considering the scarce samples (67 runs). BE simulations, on



(a) R-RMSE



(b) R-MD

Fig. 7. Difference between CMT-nek validation runs and multi-fidelity predictions based on CMT-bone

the other hand, had all the 125 samples in the design space and thus, lead to better MFS predictions. This is supported by residual errors of $\hat{f}_L(\boldsymbol{x})$ from Table 1.

In all three cases, the error in prediction ended less than 10%; thus, proving it to be a valuable approach to use for reducing computational budget in the process of co-design. The range of scale factor (ρ) for the three cases are summarized in Table 2. The scale factors are around 1 which indicates that the

Table 2. Range of the scale factors for LS-MFS

	Case 1	Case 2	Case 3
Minimum ρ	0.8	0.91	0.5
Maximum ρ	0.98	1.08	1.3

LF and HF have similar trend. The major difference between HF and LF are well-compensated by the constant/linear discrepancy function.

7 Conclusions

Due to high computational cost, validation samples from HFM are usually obtained at small scale. But with MFS model, we were able to perform quantitative validation at a reduced computational budget. In this paper, we studied the least-square MFS (LS-MFS) using polynomial response surface as it is robust with noise effect. For future work, different multi-fidelity surrogates can be compared. Our ultimate goal is to predict the performance for exascale computation platform which is essentially long-range extrapolation far from the validation samples. In the future, we will investigate the capability of LS-MFS for long-range extrapolation which suffers large uncertainty. We also noticed the LS-MFS predictions were sensitive with the high-fidelity samples. Effective design of experiments for validation runs are expected to improve the accuracy of LS-MFS, which is also a valuable research direction.

Acknowledgment

This work is supported by the U.S. Department of Energy, National Nuclear Security Administration, Advanced Simulation and Computing Program, as a Cooperative Agreement under the Predictive Science Academic Alliance Program, under Contract No. DE-NA0002378.

References

1. Center for Compressible Multiphase Turbulence webpage. `https://www.eng.ufl.edu/ccmt/`, updated: 2015-02-15
2. NEK5000 webpage. `https://nek5000.mcs.anl.gov/`
3. Vulcan Supercomputer, LLNL webpage. `https://computation.llnl.gov/computers/vulcan`
4. Balabanov, V., Grossman, B., Watson, L., Mason, W., Haftka, R.: Multi-fidelity response surface model for hsct wing bending material weight. 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization p. 4804 (1998)
5. Banerjee, T., Hackl, J., Shringarpure, M., Islam, T., Balachandar, S., Jackson, T., Ranka, S.: Cmt-bone: A proxy application for compressible multiphase turbulent flows. In: 2016 IEEE 23rd International Conference on High Performance Computing (HiPC). pp. 173–182 (Dec 2016)

6. Barrett, R.F., Borkar, S., Dosanjh, S.S., Hammond, S.D., Heroux, M.A., Hu, X.S., Luitjens, J., Parker, S.G., Shalf, J., Tang, L.: On the role of co-design in high performance computing (2013)
7. Barrett, R.F., Vaughan, C.T., Heroux, M.A.: Minighost: a miniapp for exploring boundary exchange strategies using stencil computations in scientific parallel computing. Sandia National Laboratories, Tech. Rep. SAND 5294832 (2011)
8. Box, George, E.P., Stuart Hunter, J., Hunter, W.G.: Statistics for experimenters: design, innovation, and discovery. New York: Wiley-Interscience 2 (2005)
9. Dosanjh, S.S., Barrett, R.F., Doerfler, D., Hammond, S.D., Hemmert, K.S., Heroux, M.A., Lin, P.T., Pedretti, K.T., Rodrigues, A.F., T, T.: Exascale design space exploration and co-design. Future Generation Computer System 30, 46–58 (2014)
10. Dosanjh, S.S., Barrett, R.F., Doerfler, D., Hammond, S.D., Hemmert, K.S., Heroux, M.A., Lin, P.T., Pedretti, K.T., Rodrigues, A.F., T, T., et al.: Assessing the role of mini-applications in predicting key performance characteristics of scientific and engineering applications. Journal of Parallel and Distributed Computing 30, 107–122 (2014)
11. Ellis, M., Mathews, E.: A new simplified thermal design tool for architects. Building and environment 36, 1009–1021 (2011)
12. Fernández-Godino, M.G., Park, C., Kim, N.H., Haftka, R.T.: Review of multi-fidelity models. arXiv preprint arXiv:1609.07196 (2016)
13. Heroux, M.A., Doerfler, D.W., Crozier, P.S., Willenbring, J.M., Edwards, H.C., Williams, A., Rajan, M., Keiter, E.R., Thornquist, H.K., Numrich, R.W.: Improving performance via mini-applications. Sandia National Laboratories, Tech. Rep. SAND2009-5574 3 (2009)
14. Huang, D., Allen, T., Notz, W., , Miller, R.: Sequential kriging optimization using multiple-fidelity evaluations. Structural and Multidisciplinary Optimization 32, 369–382 (2006)
15. Jin, R., Chen, W., Sudjianto, A.: An efficient algorithm for constructing optimal design of computer experiments. Journal of Statistical Planning and Inference 134, 268–287 (2005)
16. Karlin, I., Keasler, J., Neely, R.: Lulesh 2.0 updates and changes. Tech. Rep. LLNL-TR-641973 (2013)
17. Kennedy, M.C., O'Hagan, A.: Bayesian calibration of computer models. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63(3), 425–464 (2001)
18. Kumar, N., Sringarpure, M., Banerjee, T., Hackl, J., Balachandar, S., Lam, H., George, A., Ranka, S.: Cmt-bone: A mini-app for compressible multiphase turbulence simulation software. In: 2015 IEEE International Conference on Cluster Computing. pp. 785–792 (Sept 2015)
19. Kumar, N., Pascoe, C., Hajas, C., Lam, H., Stitt, G., George, A.: Behavioral emulation for scalable design-space exploration of algorithms and architectures. In: International Conference on High Performance Computing. pp. 5–17. Springer (2016)
20. L, L.G.: Multi-fidelity gaussian process regression for computer experiments. Universit Paris-Diderot-Paris VII (2013)
21. Peherstorfer, B., Willcox, K., Gunzburger, M.: Survey of multifidelity methods in uncertainty propagation, inference, and optimization (2016)
22. Qian, P.Z., Wu, C.J.: Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments. Technometrics 50, 192–204 (2008)
23. Xiong, Y., Chen, W., Tsui, K.L.: A new variable-fidelity optimization framework based on model fusion and objective-oriented sequential sampling. Journal of Mechanical Design 130(11), 111401 (2008)

24. Zhang, Y., Kim, N.H., Park, C., Haftka, R.T.: Multi-fidelity surrogate based on single linear regression. ArXiv e-prints 1705 (2017)
25. Zhang, Y., Meeker, J., Schutte, J., Kim, N., Haftka, R.: On approaches to combine experimental strength and simulation with application to open-hole-tension configuration. In: Proceedings of the American Society for Composites: Thirty-First Technical Conference (2016)
26. Zheng, L., Hendrick, T.L., Mittal, R.: A multi-fidelity modelling approach for evaluation and optimization of wing stroke aerodynamics in flapping flight. Journal of Fluid Mechanics 721, 118–154 (2013)