UNIVERSITY OF CALIFORNIA,
IRVINE

Multi-fidelity Data Fusion with Uncertainty Quantification

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Mechanical and Aerospace Engineering

by

Carlos Mora Sardiña

Thesis Committee:
Professor Ramin Bostanabad, Chair
Professor Diran Apelian
Professor Penghui Cao

2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to my supervisor, Professor Ramin Bostanabad, for providing me with unwavering support and guidance. His expertise and dedication have greatly contributed to my professional and personal development. I would also like to extend my gratitude to my lab mates for their teamwork and valuable insights, which have significantly enhanced my research experience.

I would like to thank Professor Diran Apelian and Professor Penghui Cao for agreeing to serve on the committee for this thesis.

I am deeply thankful to my loving family for their unconditional support and encouragement throughout all aspects of my life. I would also like to thank my dear friends from Barcelona, who have stood by me from afar, and the new friends I have made in California, with whom I have shared many unforgettable experiences during my time at the University of California, Irvine.

Finally, I would like to express my sincere appreciation to the Balsells fellowship for their financial support. This fellowship has provided me with a unique opportunity for personal and academic growth, for which I will be forever grateful.

ABSTRACT OF THE THESIS

Multi-fidelity Data Fusion with Uncertainty Quantification

By

Carlos Mora Sardiña

Master of Science in Mechanical and Aerospace Engineering

University of California, Irvine, 2023

Professor Ramin Bostanabad, Chair

In many applications in engineering and sciences analysts have simultaneous access to multiple data sources. In such cases, the overall cost of acquiring information can be reduced via data fusion or multi-fidelity (MF) modeling where one leverages inexpensive low-fidelity (LF) sources to reduce the reliance on expensive high-fidelity (HF) data. In this thesis we present two main contributions in the field of data fusion and its application in engineering. In particular, we introduce: (1) a novel neural network (NN) architecture for data fusion under uncertainty, namely Probabilistic Neural Data Fusion (Pro-NDF), and (2) a MF calibration scheme based on Latent Map Gaussian Processes (LMGPs) for fracture modelling of metallic components via reduced-order models (ROMs).

In the context of building an emulator for data fusion under uncertainty, we introduce Pro-NDF, a novel NN architecture that converts MF modeling into a nonlinear manifold learning problem. Pro-NDF inversely learns non-trivial (e.g., non-additive and nonhierarchical) biases of the LF sources in an interpretable and visualizable manifold where each data source is encoded via a low-dimensional distribution. This probabilistic manifold quantifies model form uncertainties such that LF sources with small bias are encoded close to the HF source. Through a set of analytic and engineering examples, we demonstrate that our approach provides a high predictive power while quantifying various sources of uncertainty.

In the context of fracture modelling of metallic materials with microscopic pores, we propose a data-driven framework that integrates a mechanistic ROM with a MF calibration scheme based on LMGPs. The proposed ROM offers computational speedup compared to direct numerical simulations (DNS) by solving a reduced-order representation of the governing equations and reducing the degrees of freedom via clustering. Since clustering affects local strain fields and hence the fracture response, we employ LMGPs to calibrate the damage parameters of an ROM as a function of microstructure and clustering, i.e., fidelity level, such that the ROM (i.e., LF source) faithfully emulates DNS (i.e., HF source). We demonstrate the application of our MF framework in predicting the damage behavior of a multiscale metallic component with spatially varying porosity. Our results indicate that microstructural porosity can significantly affect the performance of macro-components and hence must be considered in the design process.

Chapter 1

Introduction

In an increasing number of applications in engineering and sciences analysts have simultaneous access to multiple sources of information. For instance, material properties can be estimated via multiple techniques such as (in decreasing order of cost and accuracy/fidelity) experiments, direct numerical simulations (DNS), a host of physics-based reduced order models (ROMs), or analytical methods [1, 2, 3, 4, 5]. In such applications, the overall cost of gathering information about the system of interest can be reduced via *multi-fidelity* (MF) *modeling* or *data fusion* where one leverages inexpensive low-fidelity (LF) sources to reduce the reliance on expensive high-fidelity (HF) data sources. Over the past few decades, many techniques have been developed for building MF surrogates which are used in outer-loop applications such as design optimization [6, 7], calibration of computer models [8], or Bayesian optimization [9]. The main motivation behind these techniques is to leverage the correlations between LF and HF data sources (and the fact that sampling from the former is typically cheaper) to improve the predictive performance of the surrogate while reducing the overall data acquisition costs. Within the context of data fusion and its application in engineering, we have organized the contributions of this thesis into two parts.

In Chapter 2, we introduce a novel NN architecture for data fusion in scenarios where data is scarce and obtained from multiple sources with varying levels of fidelity and cost (i.e., data is unbalanced since more samples are available from cheaper sources). The proposed approach not only facilitates MF modeling, but also quantifies and visualizes the discrepancies/similarities between all data sources. In addition to that, we illustrate that a Bayesian treatment, besides alleviating overfitting and providing a probabilistic surrogate (i.e., an emulator), provides the means to develop a novel loss function (based on proper scoring rules) that improves the performance and robustness of the resulting MF NN emulator. As we demonstrate through a host of analytic and engineering examples, our approach provides a high predictive power while quantifying various sources of uncertainty, without relying on any prior knowledge of the hierarchy between the sources. Our codes and examples can be accessed via GitLab[1].

In Chapter 3, we present an engineering application of data fusion where we leverage an MF emulator to calibrate a reduced-order model (ROM) for multiscale damage analysis. In particular, we open the doors for the fracture-aware design of multiscale materials by proposing a data-driven framework that integrates a mechanistic ROM with a MF calibration scheme based on LMGPs. We show that the integration of these two components enables us to build calibrated multi-fidelity ROMs that can simulate the damage behavior of multiscale materials with spatially varying microstructures.

The main contributions of this thesis are discussed in Section 2.5 and Section 3.6, which are summarized in Chapter 4.

[1]GitLab repository: https://gitlab.com/TammerUCI/pro-ndf

Chapter 2

Probabilistic Neural Data Fusion for Learning from an Arbitrary Number of Multi-fidelity Data Sets

2.1 Related works and motivation

Early works in the field of data fusion focused primarily on hierarchically linking bi-fidelity data. For instance, in space mapping [10, 11, 12] or multi-level [13, 14, 15] techniques the inputs of the LF data are mapped following formulations such as $\boldsymbol{x}^l = F(\boldsymbol{x}^h)$ where $\boldsymbol{x}^l$ and $\boldsymbol{x}^h$ are the inputs of LF and HF sources, respectively. In this equation, $F(\cdot)$ is a transformation function whose predefined functional form is calibrated such that $y^l(F(\boldsymbol{x}^h))$ approximates $y^h(\boldsymbol{x}^h)$ as closely as possible. These techniques are useful in applications where higher fidelity data are obtained by successively refining the discretization in simulations [13, 14], e.g., by refining the mesh when modeling the flow around an airfoil or estimating the fracture toughness of a microstructure. The main disadvantages of space mapping techniques are that

(1) they rely on iterative and time-consuming analysis for choosing a near-optimal functional form for $F(\cdot)$, (2) they cannot jointly fuse more than two data sources at a time, (3) they quantify similarity/discrepancy between the sources based on pre-defined functions whose space may not include the true discrepancy, and (4) they do not quantify some uncertainty sources (such as lack of data) and are rarely formulated within a Bayesian setting that leverages prior information.

A well-known hierarchical bi-fidelity modeling framework is that of Kennedy and O'Hagan (KOH) [16] who assume that the discrepancy between the LF and HF sources is additive (multiplicative terms have also been explored [17]) and that both sources as well as the discrepancy between them can be modeled via Gaussian processes (GPs). Upon this modeling assumption, KOH find the joint posterior of GPs' hyperparameters via either fully [18, 19] or modular Bayesian inference [20, 21, 22, 23]. While KOH's approach considers multiple uncertainties and has been successfully applied to a broad range of applications [24, 25, 26], it has three main limitations: (1) it only accommodates two data sources at a time, (2) it places a priori independence assumption between the GPs, and (3) it does not provide a low-dimensional, visualizable, and interpretable metric that quantifies the correlations between the data sources.

Recent works have acknowledged the limitations of hierarchical methods and devised new methodologies to address them. For instance, MF modeling can be achieved via a recursive scheme [27] where a bi-fidelity method is repeatedly applied from the lowest to the highest fidelities. However, such recursive schemes inherit the limitations of bi-fidelity methods, cannot *jointly* fuse multi-source data sets, and are sensitive to the ordering (i.e., the relative accuracy of all sources must be known a priori).

As another example, [28, 29] presents MF networks (MFNets): an approach based on directed acyclic graphs that builds a MF surrogate using an arbitrary number of data sources. MFNets accommodate noisy data and are trained via gradient-based minimization of a nonlinear least

squares objective. Although MFNets can deal with many of the issues of state-of-the-art MF models, e.g., they can learn non-hierarchical relations between data sources, they: (1) rely on having prior knowledge on a set of latent variables that explain the relations between the sources, (2) assume each source can be represented via a linear subspace model, (3) are not probabilistic and also require regularization, (4) impose independence assumption among the data sources to derive the likelihood (i.e., the objective) function, and (5) rely on iterative approaches for finding the optimal graph structure.

Other notable works that have studied the limitations of hierarchical techniques include [30, 31, 32] which are focused on identifying (and correcting) non-additive discrepancies between LF and HF sources. However, the proposed solution in these works is intrusive and relies on some rather strong modeling assumptions that largely limit the applications. These limitations arise because the formulation of the discrepancy is learned via an embedded operator whose functional form and interaction with the LF source are constructed a priori.

A novel GP-based approach [33] addresses the above issues by converting MF modeling into a manifold learning problem where the relations between the sources are automatically quantified via an appropriately learnt distance measure. The conversion is achieved via Latent Map Gaussian Processes [33] (LMGPs, see Section 2.2.1) which enable GPs to handle categorical variables and, correspondingly, data fusion: by augmenting the inputs via a categorical variable (which indicates the source of a data point) and then concatenating all the data sets, LMGPs can simultaneously learn from an arbitrary number of information sources.

In this work , we examine the potential of NNs in matching and potentially improving state-of-the-art techniques for MF modeling such as LMGPs. Our current studies are motivated by the facts that (1) when viewed as (probabilistic or deterministic) graphical models [34], NNs provide unique opportunities to use MF data sets to uncover complex hidden relations between the corresponding sources, (2) the recent hardware and software advancements have

drastically accelerated architecture design and training of NNs, and (3) NNs scale to higher dimensions and big data significantly better than GPs.

Over the past few years some NN-based approaches have been developed for MF modeling [35, 36, 37, 38]. However, most of these works design the network architecture primarily based on hierarchical methods and consequently inherit their limitations. For instance, [35] builds two sequentially connected deterministic networks based on KOH's method where the first and second NNs are tasked to emulate the LF and HF sources, respectively. In addition to sharing the limitations of KOH's method, such a sequential bi-fidelity NN requires that the two parts of the network are trained separately (unless the LF and HF training data are available at the same inputs) and also relies on manual tuning of the architecture and loss function. Similarly, [36] build a hierarchical bi-fidelity method that allows all-at-once training of the two sequentially connected NNs (regardless of data location) but their approach is also inspired by that of KOH and requires iterative fine tuning of the architecture and loss function. It has been argued [37] that such sequentially trained NNs bridge MF modeling with transfer learning where the knowledge gained from the LF data is used in building the NN module that surrogates the HF source.

Non-sequential NNs are rarely used for MF modeling (especially with more than 2 sources) due to the fact that searching for the optimum architecture (and effectively training it with small data) is a difficult task. We address this challenge by drawing inspiration from LMGPs where we design the architecture such that any number of MF data sets can be simultaneously fused and the overall discrepancies between sources are quantified with visualizable metrics. We also illustrate that making specific parts of the network probabilistic, in addition to being superior to both deterministic and all-probabilistic NNs, enables us to infuse a proper scoring rule [39] into the loss function and, in turn, improve the performance of the MF emulator. The particular rule that we adopt is the negatively oriented interval score which is frequently used in testing the quality of probabilistic predictions but, to the best of our

knowledge, has never been used in the training stage of a probabilistic NN. In summary, our major contributions are as follows:

- We introduce a unique NN architecture for MF modeling that can fuse an arbitrary number of data sets and quantify both epistemic and aleatoric uncertainties.
- We inversely learn the accuracy of the LF sources (with respect to the HF source) and visualize the learned relations in an interpretable manifold.
- We show that a probabilistic setting allows us to develop a novel loss function (based on proper scoring rules) that improves the performance of the emulator.
- We validate the performance of our approach on analytical and real-world examples and show that it performs on par with the state-of-the-art while providing improved scalability to high dimensions and big data.

The rest of this chapter is organized as follows. We review the relevant technical background in Section 2.2 and then introduce our approach in Section 2.3. We test the performance of our approach on a host of analytical problems and real-world data sets in Section 2.4 and we draw conclusions in Section 2.5.

2.2 Technical Background

In this section we first review LMGPs which are extensions of GPs that handle categorical inputs and, thus, can readily fuse any number of data sets. Then, we provide some background on Bayesian neural networks (BNNs) which form the foundation of our neural data fusion framework.

2.2.1 Latent Map Gaussian Processes (LMGPs)

Let us denote the categorical inputs by $\boldsymbol{t} = [t_1, \ldots, t_{dt}]^T$ where the total number of distinct levels for qualitative variable t_i is τ_i. To handle mixed inputs, LMGP learns a parametric function that maps categorical variables to some points in a quantitative manifold or latent space[1]. These points (and hence the mapping function) can be incorporated into any standard correlation function, such as the Gaussian, which is reformulated as follows for mixed inputs:

$$r\left((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}^{'},\boldsymbol{t}^{'})\right) = \exp\left(-\left\|\boldsymbol{z}(\boldsymbol{t})-\boldsymbol{z}(\boldsymbol{t}^{'})\right\|_2^2 - \left(\boldsymbol{x}-\boldsymbol{x}^{'}\right)^T 10^{\boldsymbol{\Omega}}\left(\boldsymbol{x}-\boldsymbol{x}^{'}\right)\right) \tag{2.1}$$

or, equivalently,

$$r\left((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}^{'},\boldsymbol{t}^{'})\right) = \exp\left(-\sum_{i=1}^{dx} 10^{\omega_i}(x_i - x_i^{'})^2\right) \times \exp\left(-\sum_{i=1}^{dz}(z_i(\boldsymbol{t}) - z_i(\boldsymbol{t}^{'}))^2\right) \tag{2.2}$$

where $\|\cdot\|_2$ denotes the Euclidean 2-norm and $\boldsymbol{z}(\boldsymbol{t}) = [z_1(\boldsymbol{t}), \ldots, z_{dz}(\boldsymbol{t})]_{1\times dz}$ is the to-be-learned latent space point corresponding to the particular combination of categorical variables denoted by $\boldsymbol{t}$. To find these points in the latent space, LMGP assigns a unique vector (i.e., a prior representation) to each combination of categorical variables. Then, it uses matrix multiplication[2] to map each of these vectors to a point in a latent space of dimension dz:

$$\boldsymbol{z}(\boldsymbol{t}) = \boldsymbol{\zeta}(\boldsymbol{t})\boldsymbol{A} \tag{2.3}$$

where $\boldsymbol{\zeta}(\boldsymbol{t})$ is the $1 \times \sum_{i=1}^{dt} \tau_i$ unique prior vector representation of $\boldsymbol{t}$ and $\boldsymbol{A}$ is a $\sum_{i=1}^{dt} \tau_i \times dz$ matrix that maps $\boldsymbol{\zeta}(\boldsymbol{t})$ to $\boldsymbol{z}(\boldsymbol{t})$. In this work, we use $dz = 2$ since it simplifies visualization and has been shown to provide sufficient flexibility for learning the latent relations [40]. We construct $\boldsymbol{\zeta}$ via a form of one-hot encoding where we first construct the $1 \times \tau_i$ vector

[1] Multiple mapping functions can also be used to build multiple manifolds. We leverage this in Section 2.4 where we build two manifolds for data fusion problems with categorical or mixed inputs.

[2] More complex transformations based on, e.g., NNs, may also be used, although we do not do so in this work.

$\boldsymbol{v}^i = \left[v_1^i, v_2^i, \ldots, v_{\tau_i}^i\right]$ for each categorical variable t_i such that $v_j^i = 1$ when t_i is at level $k = j$ and $v_j^i = 0$ when t_i is at level $k \neq j$ for $k \in 1, 2, \ldots, \tau_i$. Then, we set $\boldsymbol{\zeta}(\boldsymbol{t}) = [\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots, \boldsymbol{v}^{d_t}]$. For example, for the two categorical variables t_1 and t_2 with 2 and 3 levels, $\boldsymbol{\zeta}(\boldsymbol{t}) = [0, 1, 0, 1, 0]$ encodes the combination where both variables are at level 2.

To train an LMGP, we use maximum likelihood estimation (MLE) to jointly estimate all of its parameters:

$$\left[\hat{m}, \hat{s}^2, \hat{\boldsymbol{\omega}}, \hat{\boldsymbol{A}}\right] = \underset{m, s^2, \boldsymbol{\omega}, \boldsymbol{A}}{\operatorname{argmax}} \quad \left|2\pi s^2 \boldsymbol{R}\right|^{-\frac{1}{2}} \times \exp\left(-\frac{1}{2}(\boldsymbol{y} - \mathbf{1}m)^T (s^2 \boldsymbol{R})^{-1} (\boldsymbol{y} - \mathbf{1}m)\right) \tag{2.4}$$

where $|\cdot|$ denotes the determinant operator, $\boldsymbol{y} = [y_1, \ldots, y_n]^T$ is the $n \times 1$ vector of outputs in the training data, $\boldsymbol{R}$ is the $n \times n$ correlation matrix with the $(i, j)^{\text{th}}$ element $R_{ij} = r\left((\boldsymbol{x}^{(i)}, \boldsymbol{t}^{(i)}), (\boldsymbol{x}^{(j)}, \boldsymbol{t}^{(j)})\right)$ for $i, j = 1, \ldots, n$, and $\mathbf{1}$ is a $n \times 1$ vector of ones.

After estimating the hyperparameters, we use the conditional distribution formulas to predict the response distribution at the arbitrary point $\boldsymbol{p}^* = (\boldsymbol{x}^*, \boldsymbol{t}^*)$. The mean and variance of this normal distribution are:

$$\mathbb{E}\left[y\left(\boldsymbol{p}^*\right)\right] = \hat{m} + \boldsymbol{r}^T\left(\boldsymbol{p}^*\right) \boldsymbol{R}^{-1} \left(\boldsymbol{y} - \mathbf{1}\hat{m}\right) \tag{2.5}$$

$$\operatorname{cov}\left(y(\boldsymbol{p}^*), y(\boldsymbol{p}^{'})\right) = \hat{s}^2 r(\boldsymbol{p}^*, \boldsymbol{p}^{'}) = \hat{s}^2 \left\{1 - \boldsymbol{r}^T(\boldsymbol{p}^*) \boldsymbol{R}^{-1} \boldsymbol{r}(\boldsymbol{p}^{'}) + g(\boldsymbol{p}^*)(\mathbf{1}^T \boldsymbol{R}^{-1} \mathbf{1})^{-1} g(\boldsymbol{p}^{'})\right\} \tag{2.6}$$

where $\mathbb{E}$ denotes expectation, $\boldsymbol{r}\left(\boldsymbol{p}^*\right)$ is an $(n \times 1)$ vector with the i^{th} element $r\left(\boldsymbol{p}^{(i)}, \boldsymbol{p}^*\right)$, and $g\left(\boldsymbol{p}^*\right) = 1 - \mathbf{1}^T \boldsymbol{R}^{-1} \boldsymbol{r}\left(\boldsymbol{p}^*\right)$.

To perform data fusion via LMGP, we re-frame multi-fidelity modeling as a manifold learning problem. Assume that we have ds data sources whose inputs and outputs are denoted by $\boldsymbol{x}^{s_i}, y^{s_i}$, respectively, with $i = 1, \ldots, ds$. We first pre-process the data by appending the inputs with a single categorical variable t^s with ds levels (hereafter referred to as the source index variable) that distinguishes the data sources. Specifically, we add t^s at level i for source

s_i, i.e., $\boldsymbol{x}^{s_i} \rightarrow [\boldsymbol{x}^{s_i}, \boldsymbol{i}_{n^{s_i}\times 1}]$, where $\boldsymbol{i}_{n^{s_i}\times 1}$ is an $n^{s_i} \times 1$ vector of i's and n^{s_i} is the number of data points for source s_i. We then combine the data for all sources into one unified data set and fit an LMGP directly to it, i.e., we fit LMGP to *all* of the data from *all* sources at once.

The fitted LMGP can provide predictions for any desired data source based on the level used for t^s and as such is an emulator for all of the data sources. Additionally, since the data sources are distinguished via a categorical variable, LMGP learns the correlations between them via a visualizable latent representation and uses these correlations to improve its predictions [33]. In the case that the raw inputs contain categorical variables $\boldsymbol{t}^c$, we use separate mappings for t^s and $\boldsymbol{t}^c$, i.e., we assign unique priors $\boldsymbol{\zeta}(t^s)$ and $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ which LMGP uses to find mapping matrices $\boldsymbol{A}^s$ and $\boldsymbol{A}^c$. The latent points corresponding to each mapping are then $\boldsymbol{z}^s$ and $\boldsymbol{z}^c$, respectively.

Note that the correlation function in Equation (2.2) depends directly on the euclidean distance between a pair of latent points. This means that relative distances in the latent space directly correspond to correlations, e.g., if a pair of data sources y^{s_1} and y^{s_2} have corresponding latent points with a distance Δ in the latent space then this directly implies by Equation (2.2) that LMGP has found those two sources to have a correlation of $\exp(-\Delta^2)$.

2.2.2 Bayesian Neural Networks

Feedforward neural networks (FFNNs) are one of the most common models used in deep learning and their main goal is to learn the underlying function $f(\boldsymbol{x})$ that maps the inputs $\boldsymbol{x}$ to the target y [41]. To this end, an FFNN defines the mapping $\hat{f}(\boldsymbol{x};\boldsymbol{\theta})$ whose parameters $\boldsymbol{\theta}$ are estimated such that $\hat{y} = \hat{f}(\boldsymbol{x};\boldsymbol{\theta})$ best approximates $f(\boldsymbol{x})$. NN-based approaches for MF emulation can provide attractive advantages since they are universal function approximators [42] and can handle high-dimensional inputs and large data sets. In this subsection, we first briefly describe the working principle of FFNNs and then motivate the use of BNNs and

Bayes by backprop [43].

FFNNs propagate information from the inputs $\boldsymbol{x}$ to the output y through intermediate computations that define $\hat{f}$. They are traditionally built via a succession of L layers where $L-2$ hidden layers are placed between the input and output layers. The output of layer k is denoted by $\boldsymbol{z}_k$ and is obtained as follows:

$$\boldsymbol{z}_1 = \boldsymbol{x}, \tag{2.7}$$

$$\boldsymbol{z}_k = \phi_k \left(\boldsymbol{W}_k \boldsymbol{z}_{k-1} + \boldsymbol{b}_k \right) \quad \forall\ k \in [2, L-1], \tag{2.8}$$

$$\hat{y} = \phi_L \left(\boldsymbol{W}_L \boldsymbol{z}_{L-1} + b_L \right) \tag{2.9}$$

where ϕ is the (typically non-linear) activation function. The parameters $\boldsymbol{\theta}_k = (\boldsymbol{W}_k, \boldsymbol{b}_k)$, where $\boldsymbol{W}_k$ and $\boldsymbol{b}_k$ are the weight matrices and bias vectors, respectively, correspond to the connections between the $(k-1)^{th}$ and k^{th} layer. For brevity, we denote the parameters of the entire network by $\boldsymbol{\theta}$.

Since $\hat{f}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta})$ approximates $y^{(i)}$, we can write:

$$y^{(i)} = f(\boldsymbol{x}^{(i)}) + \epsilon \approx \hat{f}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}) + \epsilon \tag{2.10}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ represents noise. Equation (2.10) indicates that $P(y^{(i)}|\boldsymbol{x}^{(i)}) \sim \mathcal{N}(f(\boldsymbol{x}^{(i)}), \sigma^2)$ or $\mathbb{E}[y^{(i)}|\boldsymbol{x}^{(i)}] = f(\boldsymbol{x}^{(i)}) \approx \hat{f}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta})$. Therefore, FFNNs can be seen as a statistical model with parameters $\boldsymbol{\theta}$ that aim to learn the expected conditional distribution $\mathbb{E}[y^{(i)}|\boldsymbol{x}^{(i)}]$. To that end, the conditional probability $P(\mathcal{D}|\boldsymbol{\theta})$ is written as:

$$\begin{aligned} P(\mathcal{D}|\boldsymbol{\theta}) &= \prod_{i=1}^{n} P(y^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta}) P(\boldsymbol{x}^{(i)}) = \prod_{i=1}^{n} \mathcal{N}(y^{(i)}; \hat{y}^{(i)}, \sigma^2) P(\boldsymbol{x}^{(i)}) \\ &= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} \left(y^{(i)} - \hat{y}^{(i)} \right)^2 \right) P(\boldsymbol{x}^{(i)}) \end{aligned} \tag{2.11}$$

Since the likelihood function $L(\boldsymbol{\theta}) \equiv P(\mathcal{D}|\boldsymbol{\theta})$, the parameters $\boldsymbol{\theta}$ can be estimated by maximizing $L(\boldsymbol{\theta})$ (the dependence on $\boldsymbol{x}$ is dropped for brevity):

$$\boldsymbol{\theta}_{MLE} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) \equiv \arg\min_{\boldsymbol{\theta}} \; -\log P(\mathcal{D}|\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - \hat{f}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}) \right)^2 \quad (2.12)$$

which is equivalent to minimizing the mean squared error (MSE) of the predictions $\hat{y} = \hat{f}(\boldsymbol{x}; \boldsymbol{\theta})$ with respect to the targets y. Equation (2.12) can be updated via Bayes rule to consider prior knowledge on $\boldsymbol{\theta}$ in the optimization. These maximum a posteriori (MAP) estimates are obtained via:

$$\boldsymbol{\theta}_{MAP} = \arg\max_{\boldsymbol{\theta}} P\left(\boldsymbol{\theta}|\mathcal{D}\right) = \arg\max_{\boldsymbol{\theta}} \log P\left(\boldsymbol{\theta}|\mathcal{D}\right) = \arg\max_{\boldsymbol{\theta}} \left(\log P\left(\mathcal{D}|\boldsymbol{\theta}\right) + \log P(\boldsymbol{\theta}) \right) \quad (2.13)$$

where the first term recovers MSE as in Equation (2.12) and the second term depends on the prior distribution assigned to the parameters. Equation (2.13) illustrates that Gaussian and Laplacian priors are equivalent to $L2$ and $L1$ regularization, respectively [41, 43].

FFNNs are likely to overfit in scenarios where data is scarce. Additionally, they cannot directly quantify prediction uncertainty and are often overconfident in extrapolation [44]. BNNs are developed to address these issues [45, 46]. In BNNs, the weights are endowed with probability distributions (rather than single point estimates) which naturally results in probabilistic predictions and can dramatically reduce overfitting via parameter regularization and model averaging.

Predictions via a BNN requires sampling from the posterior distribution of the parameters, i.e., $P\left(\boldsymbol{\theta}|\mathcal{D}\right)$, which does not have a closed form and is highly complex. Over the past few years, various techniques have been developed to obtain samples from $P\left(\boldsymbol{\theta}|\mathcal{D}\right)$ (or an approximation thereof). The most popular techniques are based on either Markov Chain Monte Carlo (MCMC) [47] or variational inference (VI) [48] which, unlike MCMC, learns an approximation of the posterior distribution.

Although MCMC methods are arguably the best techniques for sampling from the *exact* posterior, their lack of scalability makes them inefficient for BNNs of any practical size [49]. Hence, we employ Bayes by backprop [43] which is a variational method that approximates $P(\boldsymbol{\theta}|\mathcal{D})$ with the parameterized distribution $q(\boldsymbol{\theta}|\boldsymbol{\varphi})$. The parameters $\boldsymbol{\varphi}$ are learned by minimizing the Kullback–Leibler (KL) divergence between the true and approximated posteriors:

$$\begin{aligned}
&\mathrm{KL}[q(\boldsymbol{\theta}|\boldsymbol{\varphi})||P(\boldsymbol{\theta}|\mathcal{D})] = \int q(\boldsymbol{\theta}|\boldsymbol{\varphi}) \log \left(\frac{q(\boldsymbol{\theta}|\boldsymbol{\varphi})}{P(\boldsymbol{\theta}|\mathcal{D})} \right) d\boldsymbol{\theta} = \int q(\boldsymbol{\theta}|\boldsymbol{\varphi}) \log \left(\frac{q(\boldsymbol{\theta}|\boldsymbol{\varphi}) P(\mathcal{D})}{P(\mathcal{D}|\boldsymbol{\theta}) P(\boldsymbol{\theta})} \right) d\boldsymbol{\theta} \\
&= \int q(\boldsymbol{\theta}|\boldsymbol{\varphi}) \log P(\mathcal{D}) d\boldsymbol{\theta} + \int q(\boldsymbol{\theta}|\boldsymbol{\varphi}) \log \left(\frac{q(\boldsymbol{\theta}|\boldsymbol{\varphi})}{P(\boldsymbol{\theta})} \right) d\boldsymbol{\theta} - \int q(\boldsymbol{\theta}|\boldsymbol{\varphi}) \log P(\mathcal{D}|\boldsymbol{\theta}) d\boldsymbol{\theta} \\
&= \log P(\mathcal{D}) + \mathrm{KL}[q(\boldsymbol{\theta}||\boldsymbol{\varphi})|P(\boldsymbol{\theta})] - \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\varphi})}[\log P(\mathcal{D}|\boldsymbol{\theta})] \qquad (2.14)
\end{aligned}$$

where Bayes rule is applied to $P(\boldsymbol{\theta}|\mathcal{D})$ in the first line. Then, the parameters $\boldsymbol{\varphi}$ are estimated by minimizing Equation (2.14):

$$\boldsymbol{\varphi}^* = \underset{\boldsymbol{\varphi}}{\operatorname{argmin}} \, \mathrm{KL}[q(\boldsymbol{\theta}|\boldsymbol{\varphi})||P(\boldsymbol{\theta}|\mathcal{D})] = \underset{\boldsymbol{\varphi}}{\operatorname{argmin}} \, \mathrm{KL}[q(\boldsymbol{\theta}|\boldsymbol{\varphi})||P(\boldsymbol{\theta})] - \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\varphi})}[\log P(\mathcal{D}|\boldsymbol{\theta})] \quad (2.15)$$

where the term $\log P(\mathcal{D})$ is excluded as it is constant. Equation (2.15) aims to minimize the sum of two terms. The second term corresponds to the expectation of the negative log-likelihood while the first term acts as a regularizer and corresponds to the KL divergence between the approximated posterior and the prior.

2.3 Probabilistic Neural Data Fusion

Designing a multi-fidelity NN that leverages an ensemble of LF data sets to better learn an HF source is a very challenging task because of the following major reasons:

1. The relations among the data sources can be unknown. For instance, in the Rational

example (see Table A.2 in Appendix A.1) there are three LF sources whose biases are *not* additive. Additionally, these LF sources are not hierarchically ordered in the sense that the second LF source is more accurate than the first one.

2. There are typically (but not always) more LF data available since LF sources are generally cheaper compared to the HF source. Learning from such an unbalanced MF data is quite difficult especially in the presence of scarce HF data (as an example, see the sample sizes for the engineering applications described in Appendix A.2).

3. NNs can be built in many ways and, as shown in Section 2.4, their performance heavily depends on their architecture and training mechanism. Building an optimum[3] NN with small, unbalanced, and MF data is even more difficult since the sensitivity to the architecture and training mechanism considerably increases.

We propose to address the above challenges by converting MF modeling to a manifold[4] learning problem which is then solved via an NN. We design the architecture, loss function, and training mechanism of this NN with a particular focus on uncertainty sources that include data scarcity (especially HF samples), noise with unknown variance (which can affect any of the data sources), non-trivial biases of LF sources, and data imbalances.

As schematically demonstrated in Figure 2.1, we convert MF modeling to manifold learning by augmenting the input space with the categorical variable t^s whose levels (e.g., $\{'1', '2', \cdots\}$ or $\{a, b, \cdots\}$) indicate the source that generates a sample. We then map this *source indicator* variable to a low-dimensional manifold via a BNN (see Block 1 in Figure 2.1). If the original input space has the categorical variables $\boldsymbol{t}^c$, we similarly map them to a manifold (but this time we use a deterministic NN, see Block 2 in Figure 2.1). Afterwards, we combine the latent variables of these two manifolds with the quantitative inputs $\boldsymbol{x}$ via a deterministic NN, see Block 3 in Figure 2.1. As opposed to the other two blocks, we require Block 3 to produce

[3] We measure optimality in terms of NN's error in predicting unseen data from the HF source.

[4] A manifold or a latent-space is a compact representation of a high-dimensional object such as an image.

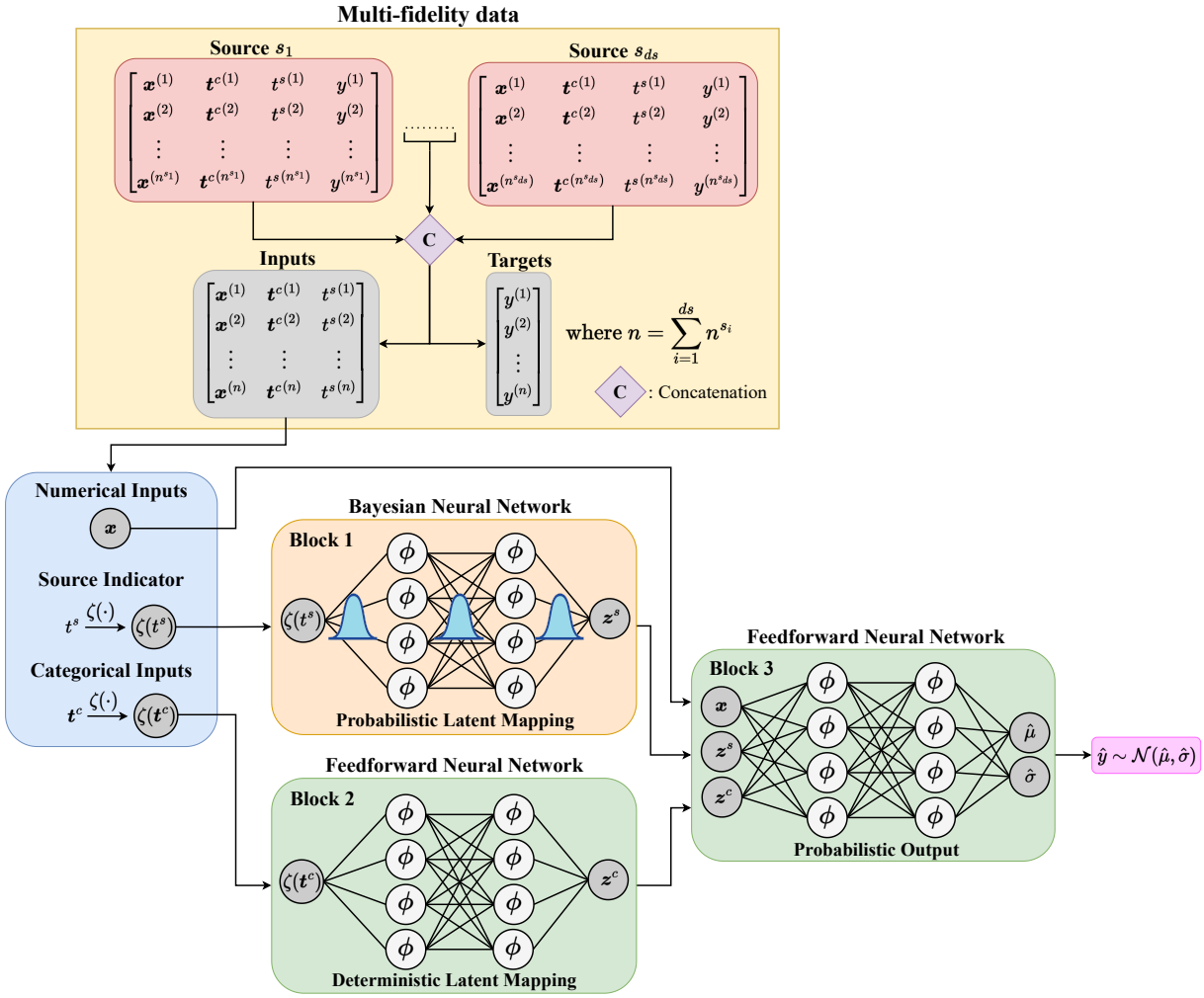

Figure 2.1 Probabilistic neural data fusion (Pro-NDF): The proposed architecture allows to combine an arbitrary number of sources by appending a source indicator variable to the data sets and then concatenating them. Pro-NDF consists of three blocks that perform separate tasks related to MF modeling: (1) Block 1 is a BNN that maps a quantitative prior representation of the source indicator $\boldsymbol{\zeta}(t^s)$ to a continuous manifold, (2) Block 2 is an FFNN that maps a quantitative prior representation of the categorical inputs $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ to a continuous manifold, and (3) Block 3 is an FFNN with a probabilistic output that maps the numerical inputs and the latent variables to a parametric distribution.

a normal probability distribution in order to capture aleatoric uncertainties. Finally, we train the entire network on the entire[5] data using our custom loss function that noticeably improves the prediction intervals.

[5]By entire, we mean the combined data sets from all sources.

In the following subsections, we elaborate on our rationale for designing a multi-block architecture and a custom loss function in Section 2.3.1 and Section 2.3.2, respectively. Then, we provide some details on the training and inference stages in Section 2.3.3.

2.3.1 Multi-Block Architecture

Each block of our network is designed to address particular challenges associated with MF modeling. Specifically, the BNN of Block 1 maps a quantitative prior representation $\boldsymbol{\zeta}(t^s)$ of the source indicator variable t^s to a continuous manifold $\boldsymbol{z}^s$. We design $\boldsymbol{\zeta}(t^s)$ by one-hot encoding t^s to merely inform the network about the source that generates a sample[6]. We build $\boldsymbol{z}^s$ based on a categorical variable because it forces the manifold to uncover the relations between sources (i.e., the levels of t^s). These relations are represented as distances in $\boldsymbol{z}^s$ where sources that produce similar data are encoded with close-by points (see Section 2.4 for multiple examples). This distance learning is in sharp contrast to existing approaches since (1) it does not assume there is any hierarchy between the data sources, (2) it is scalable to an arbitrary number of data sets, (3) it enables training the entire network via all available samples, (4) it is visualizable and interpretable which helps in identifying anomalous data sources, and (5) it does not assume any specific form (e.g., additive, multiplicative, etc.) for the biases of LF sources.

Block 1 is the only part of our network where the weights and biases are endowed with probability distributions. We make this choice to better learn model form errors and more accurately quantify the epistemic uncertainties due to lack of data and source-wise discrepancies. We note that, while the outputs of Block 1 do *not* parameterize a probability distribution, they are probabilistic by nature since they are obtained by propagating the deterministic vector $\boldsymbol{\zeta}(t^s)$ through some probabilistic hidden layers.

[6]If there is some prior knowledge about the relation among the sources, $\boldsymbol{\zeta}(t^s)$ can be designed to reflect it. We do not pursue designing such informative priors in this work.

Block 2 is an FFNN that maps the quantitative prior representation $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ of the categorical inputs $\boldsymbol{t}^c$ to the manifold $\boldsymbol{z}^c$ (Block 2 is omitted if the original inputs are purely quantitative). Similar to Block 1, we design $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ via one-hot encoding and use deterministic outputs. However, unlike Block 1 we use a deterministic FFNN in Block 2 to map $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ into $\boldsymbol{z}^c$. We make this decision to reduce the number of parameters and also because the meaning (and hence effects) of categorical inputs across different sources is typically the same[7].

We set the manifold dimension to 2 for both Block 1 and Block 2, i.e., $dz^s = dz^c = 2$. While higher dimensions provide more learning capacities, our results in Section 2.4 and those reported elsewhere [50, 51, 52, 53, 54, 55] indicate that low-dimensional manifolds are quite powerful in learning highly complex relations. For instance, [56] shows that a single latent variable can encode *smiling* in images of human faces which is a high-dimensional and complex feature in the original data space. Additionally, our choice simplifies the visualization of the manifolds and reduces the chances of overfitting since we are primarily interested in scarce data applications.

Block 3 is also an FFNN that maps the numerical inputs and the latent variables in both manifolds to a parametric distribution which represents the output. Block 3 has deterministic weights and biases since source-wise uncertainties are propagated to it via Block 1. However, we equip Block 3 with a probabilistic output [57] because it: (1) quantifies aleatoric uncertainties that are inherent to the data sets[8], and (2) enables designing a multi-task loss that considers the quality of the prediction intervals (detailed in Section 2.3.2). Additionally, Block 3 is responsible for learning the behavior for all data sources simultaneously, which allows it to leverage correlations between sources to augment predictions through a process akin to weight sharing.

[7]Due to severe discrepancies such as large model form errors, the effects of a categorical variable on the response may be quite different across the sources.

[8]The predicted variance also includes epistemic uncertainties that are propagated from Block 1, see Section 2.3.3.

2.3.2 Uncertainty-Aware Loss

NNs typically provide overconfident predictions especially when they are trained on small and unbalanced data. As explained in Section 2.3.1, we aim to address this issue by making Block 1 and the network's final output probabilistic. However, for these measures to work, we must develop an effective optimization[9] scheme where the loss function appropriately rewards prediction intervals (PIs) that are sufficiently wide (but not too wide) to cover unseen data (especially HF data). To design such a loss function, we draw inspiration from strictly proper scoring rules [39] and augment Equation (2.15) with the negatively oriented interval score. Our loss is defined as:

$$\mathcal{L} = \mathcal{L}_{NLL} + \alpha_1 \mathcal{L}_{KL} + \alpha_2 \mathcal{L}_{IS} + \alpha_3 \mathcal{L}_2 \tag{2.16}$$

where $\mathcal{L}_{NLL}$ refers to the negative log-likelihood, $\mathcal{L}_{KL}$ is the KL divergence between the prior and the variational posterior distributions on the parameters (only applicable for the BNN from Block 1), $\mathcal{L}_{IS}$ denotes the interval score term, and $\mathcal{L}_2$ is $L2$ regularization (only applicable for deterministic NNs, i.e., Block 2 and 3). α_1, α_2 and α_3 are hyperparameters that, respectively, determine the relative strengths of $\mathcal{L}_{KL}$, $\mathcal{L}_{IS}$ and $\mathcal{L}_2$ compared to $\mathcal{L}_{NLL}$. The four terms in Equation (2.16) are calculated as:

$$\mathcal{L}_{NLL} = -\frac{1}{N}\sum_{i=1}^{N} \log \mathcal{N}(y^{(i)}; \hat{\mu}^{(i)}, \left(\hat{\sigma}^{(i)}\right)^2) \tag{2.17}$$

$$\mathcal{L}_{KL} = \mathrm{KL}[q(\boldsymbol{\theta}|\boldsymbol{\varphi})||P(\boldsymbol{\theta})] \tag{2.18}$$

$$\mathcal{L}_{IS} = \frac{1}{N}\sum_{i=1}^{N}[(\hat{u}^{(i)} - \hat{l}^{(i)}) + \frac{2}{\gamma}(\hat{l}^{(i)} - y^{(i)})\mathbb{1}\{y^{(i)} < \hat{l}^{(i)}\} + \frac{2}{\gamma}(y^{(i)} - \hat{u}^{(i)})\mathbb{1}\{y^{(i)} > \hat{u}^{(i)}\}] \tag{2.19}$$

$$\mathcal{L}_2 = |\boldsymbol{\theta}|^2 \tag{2.20}$$

[9]Recall that we use Bayes by backprop which takes a variational approach towards finding the posteriors, see Section 2.2.2.

where $\mathcal{L}_{KL}$ is computed via a Monte Carlo approximation, N is the batch size, and $\mathbb{1}\{\cdot\}$ denotes the indicator function that returns 1 if the event in brackets is true and 0 otherwise. The three terms of Equation (2.16) compose a multi-task loss where: (1) the likelihood term $\mathcal{L}_{NLL}$ penalizes the model if the predicted distribution does not match the target distribution, (2) the KL divergence term $\mathcal{L}_{KL}$ favors variational posteriors that are similar to the assumed prior as per Equation (2.15), and (3) the interval score term $\mathcal{L}_{IS}$ rewards narrow PIs while penalizing the model for each observation $y^{(i)}$ that lies outside the $(1-\gamma)\times 100\%$ prediction interval that spans the range $[\hat{l}^{(i)}, \hat{u}^{(i)}]$ where $\hat{l}^{(i)} = \hat{\mu}^{(i)} - 1.96\hat{\sigma}^{(i)}$ and $\hat{u}^{(i)} = \hat{\mu}^{(i)} + 1.96\hat{\sigma}^{(i)}$. In this work, we use $\gamma = 5\%$, thus implying that $\mathcal{L}_{IS}$ is minimized by a distribution whose 95% PI is as tight as possible while containing all the training data.

2.3.3 Training and Prediction

In BNNs, the variational posterior of $\boldsymbol{\theta}$ is typically defined layer-wise as a multivariate Gaussian with mean $\boldsymbol{\mu} \in \mathbb{R}^{c_k}$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{c_k \times c_k}$, i.e., $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where c_k is the total number of connections between two consecutive layers. Estimating the full covariance matrix requires learning $\mathcal{O}(c_k^2)$ parameters and is thus computationally prohibitive in most applications [49]. To reduce the costs, some simplifications have been adopted in the literature, such as learning diagonal or block diagonal [58] covariance matrices. However, our approach does not suffer from this computational issue since the only Bayesian part of our network is Block 1 (see Figure 2.1) whose size is typically very small (we use one hidden layer with 5 neurons for all the studies in Section 2.4). Hence, we estimate a dense covariance matrix between any two layers of Block 1 to improve its uncertainty quantification capacity. As for the prior, we use a zero mean Gaussian distribution with diagonal covariance matrix which makes the KL term equivalent to $L2$ regularization with a rate defined by the standard deviation of the prior distribution [59]. Thus, the standard deviation is a hyperparameter that needs to be tuned specifically to each problem.

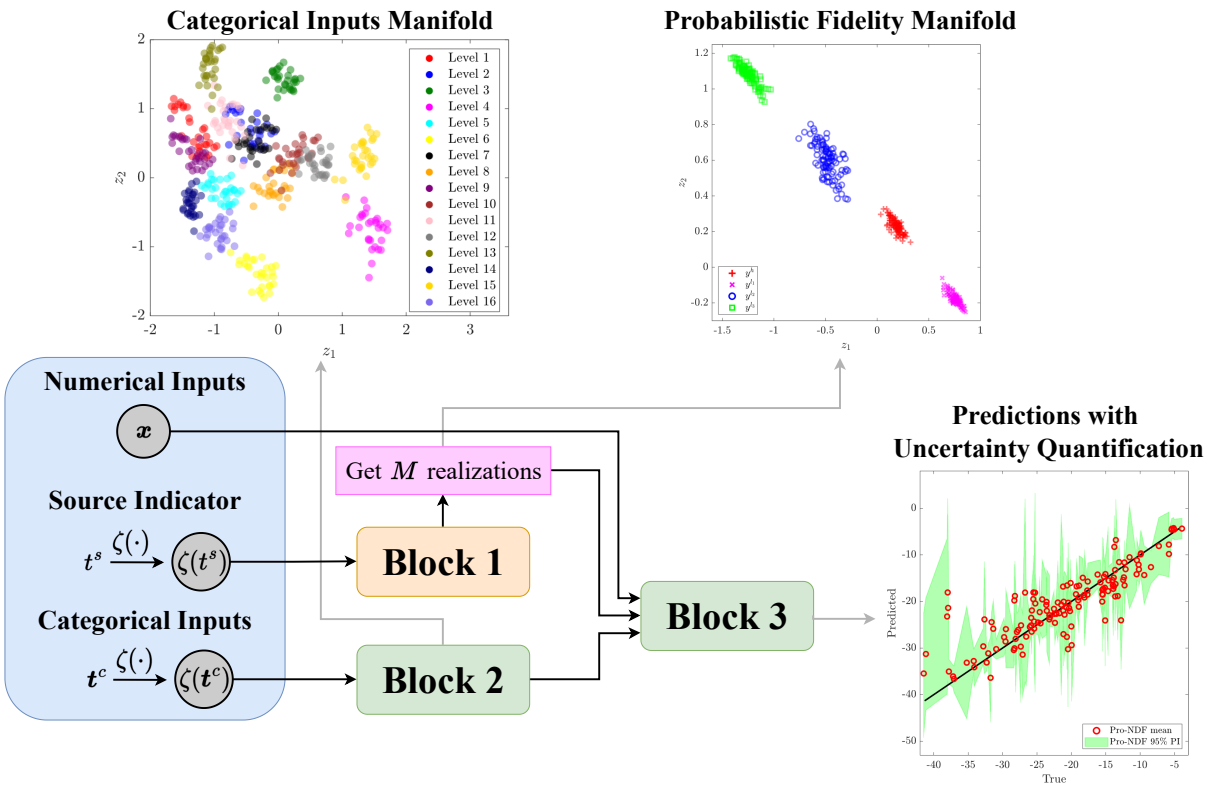

Figure 2.2 Outputs of Pro-NDF: We visualize the outputs of Pro-NDF after it is trained on the MF data of the HOIP data set, which does not have any numerical inputs (see Appendix A.2 for more details). To provide probabilistic predictions that quantify both epistemic as well as aleatoric uncertainties, Pro-NDF learns a probabilistic fidelity manifold (where sources with similar behavior are encoded with close-by distributions) and a deterministic manifold for the categorical inputs.

BNNs represent their weights and biases by parameterized distributions which in our case are multivariate normal with dense covariance matrices. In a forward pass during either training or prediction, we take individual samples from these distributions and assign them to the weights and biases. In this way, instead of explicitly obtaining the true posterior distribution of the output of Block 1 (i.e., $\boldsymbol{z}^s$, see Figure 2.1), we obtain an empirical distribution in the $\boldsymbol{z}^s$ manifold by taking a number of forward passes, see Figure 2.2. We refer to these forward passes as *realizations* and as explained below we use different number of passes in training versus prediction.

To obtain the response (in training or testing) at the input $\boldsymbol{u}$ using Pro-NDF, which contains

both a BNN component and a probabilistic output, we use ensemble prediction formulas [38]:

$$\hat{\mu}(\boldsymbol{u}) = \frac{1}{M}\sum_{j=1}^{M} \hat{\mu}_{\boldsymbol{\theta}_j}(\boldsymbol{u}) \tag{2.21}$$

$$\hat{\sigma}(\boldsymbol{u}) = \frac{1}{M}\sum_{j=1}^{M} \left(\hat{\sigma}^2_{\boldsymbol{\theta}_j}(\boldsymbol{u}) + \hat{\mu}^2_{\boldsymbol{\theta}_j}(\boldsymbol{u})\right) - \hat{\mu}^2(\boldsymbol{u}) \tag{2.22}$$

where $\hat{\mu}_{\boldsymbol{\theta}_j}(\boldsymbol{u})$ and $\hat{\sigma}_{\boldsymbol{\theta}_j}(\boldsymbol{u})$ are, respectively, the mean and standard deviation of the output distribution in the j^{th} realization and $\boldsymbol{\theta}_j$ are the associated network parameters. For predictions with a fitted NN, we use $M = 1000$ since it provides a higher accuracy in quantifying the uncertainty associated with learning the fidelity manifold (i.e., $\boldsymbol{z}^s$). While training the network, we use $M = 200$ to reduce the computational costs.

The performance of an NN is highly sensitive to its architecture and hyperparameters if the training data is small, unbalanced, and multi-fidelity. To reduce this sensitivity and leverage the low costs of training a single NN on small data, we perform automated hyperparameter tuning[10]. To this end, we use RayTune [60] and Hyperopt to find the optimum hyperparameters and architecture by minimizing the five-fold cross-validation errors on predicting the high-fidelity data.

For our approach specifically, we apply the above tuning strategy to the architecture of Block 3, the learning rate of the Adam optimizer, α_1, α_2 and α_3 in Equation (2.16), the prior standard deviation of weight matrices in Block 1, and the batch size. We fix the architectures of Block 1 and Block 2 to one hidden layer with 5 neurons and the dimension of both manifolds to 2. The activation function for all the neurons of Block 1 and 3 is hyperbolic tangent, whereas for Block 2 it is the sigmoid function. In addition to one-hot encoding the categorical inputs, we scale (normalize) the numerical inputs and outputs to the range $[0, 1]$ based on the range of the training data to assist the network in training. The

[10]We use this approach for all NN-based data fusion approaches (including ours) in Section 2.4.

outputs are un-scaled to the original space after prediction. For more information and full details on implementation, please see our GitLab repository.

2.4 Results and discussions

In this section, we validate our approach on multiple analytic and real-world MF problems (detailed in Appendix A.1 and Appendix A.2) and compare its performance against LMGP, multi-fidelity deep Gaussian processes (MF-DGP) [61], a single-fidelity GP (SF-GP) fit on the HF data, and two other existing NN-based approaches which are based on simple feedforward networks or sequential multi-fidelity (SMF) networks which are described in Appendix A.3. The hyperparameters of all the NN-based approaches are tuned as described in Section 2.3.3. We refer the reader to our GitLab repository for specific details on implementation, estimated hyperparameters, and training/test data. For LMGP, SF-GP, and MF-DGP, none of its architectural parameters (such as the kernel type, mean function, latent map, etc.) are tuned.

We first conduct an ablation study in Section 2.4.1 to quantify the impacts of our designed architecture, loss function, and probabilistic elements. Then, we test the performance of the six MF approaches on the analytic and real-world problems in Section 2.4.2 and Section 2.4.3, respectively. Finally, in Section 2.4.4 we carry out convergence studies on three bi-fidelity examples to test the performance of our approach against existing modeling techniques over a range of data set sizes. In each problem, the goal is to model the HF source as accurately as possible, i.e., to obtain the lowest mean prediction error while maximizing the number of training/test samples that fall in the 95% PI. To this end, we use normalized root mean squared-error (NRMSE) and normalized mean negatively oriented interval score (NIS)[11] as

[11]When used as evaluation metrics, the NRMSE and NIS are equivalent to Equation (A.1-1) and Equation (2.19), respectively, calculated over the test data and scaled by the standard deviation of the high-fidelity training data.

our evaluation metrics. Note that the FFNN and SMF approaches are not probabilistic, i.e., they provide point estimates rather than PIs and therefore they are only evaluated based on NRMSE.

All the experiments of this section are conducted on an NVIDIA GeForce RTX 3060 with 64 GB of RAM using NVIDIA CUDA 11.2 and cuDNN 8.4.1. We use Python 3.8.10 with Tensorflow 2.6.0, Tensorflow Probability 0.14.0, and Keras 2.6.0. We report the training time for fitting Pro-NDF and the other MF approaches in Appendix A.4.1.

2.4.1 Ablation Study

To evaluate the impact of the key components of Pro-NDF, we perform an ablation study on the Rational and DNS-ROM problems which are detailed in Appendix A.1 and Appendix A.2, respectively. Namely, we analyze the impact of:

1. Using a BNN rather than a deterministic FFNN in Block 1 for probabilistically learning the relations between the data sources.
2. Considering $\mathcal{L}_{IS}$ in the loss function of Equation (2.16).
3. Fitting the model to the parameters of a distribution instead of a scalar, i.e., using a probabilistic output.
4. Leveraging the fidelity map to detect the least accurate LF source and, in turn, assessing whether this source helps emulating the HF source.

Regarding the third item above we note that we no longer use NIS in the loss once the probabilistic output is removed. However, we still calculate the NIS after training based on the empirical distribution of the fidelity manifold which is produced by the multiple realizations of the BNN component.

Problem	Model Version	Input data				Components			NRMSE	NIS
		HF	LF1	LF2	LF3	$\mathcal{L}_{IS}$	PB1	PO		
Rational	Base	✓	✓	✓	✓	✓	✓	✓	**0.084**	**0.464**
	V1	✓	✓	✓	✓	✗	✓	✓	0.119	0.536
	V2	✓	✓	✓	✓	✓	✗	✓	0.110	0.488
	V3	✓	✓	✓	✓	✗	✓	✗	0.092	0.830
	V4	✓	✓	✓	✗	✓	✓	✓	0.156	2.021
DNS-ROM	Base	✓	✓	✓	✓	✓	✓	✓	0.101	**0.520**
	V1	✓	✓	✓	✓	✗	✓	✓	0.119	0.632
	V2	✓	✓	✓	✓	✓	✗	✓	0.155	1.186
	V3	✓	✓	✓	✓	✗	✓	✗	0.140	4.379
	V4	✓	✓	✓	✗	✓	✓	✓	**0.100**	0.564

Table 2.1 Results of the ablation study: We evaluate the effect of removing individual components of Pro-NDF from it by reporting the NRMSE and NIS on unseen HF data. All models are trained as discussed in Section 2.3 (e.g., all models benefit from automatic hyperparameter tuning). For both NRMSE and NIS, lower numbers indicate better performance. The ticks indicate whether a component is used. The acronyms and symbols are defined as: HF: high-fidelity, LF1: low-fidelity 1, LF2: low-fidelity 2, LF3: low-fidelity 3, $\mathcal{L}_{IS}$: negatively oriented interval score term in the loss function of Equation (2.16), PB1: probabilistic Block 1, PO: probabilistic output.

We summarize the results of the ablation study on the two examples in Table 2.1. For both problems, we observe that using all components minimizes the test NRMSE and NIS. Notably, both of our model's probabilistic components significantly increase the performance: the probabilistic output enables Pro-NDF to not only capture aleatoric uncertainty, but also leverage NIS in its loss function. Additionally, using a BNN improves Pro-NDF's HF emulation capabilities by preventing overfitting in scarce data regions (since Block 1 is regularized) and by partially disentangling epistemic and aleatoric uncertainties which yields better PIs.

We observe that without a probabilistic output, the NIS (and hence the uncertainty quantification accuracy) drops quite significantly (compare V3 to V1 and the base in either of the problems) since the model can no longer account for aleatoric uncertainties. By comparing V1 to the base model in either of the problems in Table 2.1 we see that for a model with a probabilistic output the optimal performance is obtained when $\mathcal{L}_{IS}$ is used in the loss. That is, leveraging the NIS in training improves both mean prediction and uncertainty quantification (measured via NRMSE and NIS, respectively).

In both problems, evaluating V1 through V3 against one another indicates that there is a trade-off between NRMSE and NIS. That is, versions that perform well in terms of NRMSE, do not generally provide the smallest NIS. However, when all of these components are included in Pro-NDF (see the base model in Table 2.1 for either of the problems), both NRMSE and NIS are reduced. This improvement is due to the fact that the priors and $\mathcal{L}_{IS}$ effectively regularize the model whose learning capacity is substantially increased by the probabilistic natures of Block 1 and the output.

The probabilistic fidelity manifold (i.e., output of Block 1) provides an intuitive and visualizable tools to learn the similarity/discrepancy among the sources. Hence, once we fit the base model in each problem, we analyze the learned fidelity manifold to determine the LF source that has the least similarity to the HF source, see Figure 2.4(a) and Figure 2.6(a). Based on the distances in the fidelity manifold of each problem, we conclude that the third LF source is the least correlated one with the HF source in both cases. We exclude this source and its data from MF modeling and refit the base model to the rest of the data, see version V4 for both problems.

One of the major outputs of Pro-NDF is the learned fidelity manifold which indicates which LF source has the highest discrepancy compared to the HF source. Hence, after training a Pro-NDF and inversely identifying the least accurate LF source, we can build another Pro-NDF while excluding the data from this source. In the Rational problem, omitting the lowest-fidelity source results in much worse NRMSE and NIS. We explain this observation by noting that this problem has an extremely small number of HF samples and therefore it is important to judiciously use all available data in training. However, in the DNS-ROM problem version V4 achieves the best NRMSE while Pro-NDF with all components achieves the best NIS and second best NRMSE (compare base to V4 in Table 2.1). We explain this trend by noting that the size of the training data in the DNS-ROM problem is significantly higher than that in the Rational problem. Therefore, omitting a highly

inaccurate data source slightly improves mean prediction accuracy for the HF source in the DNS-ROM problem since the input-output relationships learned by Block 3 for the different sources are more similar. Omitting data from this source also increases the ratio of HF data available in the unified data set which helps in learning the HF behavior. However, using all data sources provides Pro-NDF with more information which improves the uncertainty quantification capability and hence a smaller error on NIS.

2.4.2 Analytic Problems

In this section, we validate our approach against LMGP, SF-GP, MF-DGP, and two existing NN-based technologies for the Rational, Wing-weight, and Borehole examples detailed in Appendix A.1. These examples cover a wider range of input dimensionality, number of sources, and model form errors (e.g., additive and nonlinear biases). Similar to the previous section, we use NRMSE and NIS on HF test data as the performance metrics. The input space of these three examples does not have categorical features and hence both Pro-NDF and LMGP learn a single manifold. We visualize the fidelity manifolds learned by Pro-NDF and LMGP to examine these models' ability in inversely learning the relationships among the data sources (note that the LF sources are *not* ordered based on their accuracy). We highlight that MF-DGPs require knowledge of the relative fidelity of the sources and hence we provide them with this additional information. We also remark that, unlike Pro-NDF, the fidelity manifold of LMGP is not probabilistic and hence each data source is encoded with a single point in the manifold.

The results for each approach on each problem are summarized in Table 2.2 and demonstrate that the probabilistic approaches, i.e., LMGP and Pro-NDF, significantly outperform the deterministic approaches and MF-DGP (which is probabilistic) in all problems. The FFNN approach performs significantly worse than LMGP and sometimes approaches the

	Rational		Wing-weight		Borehole	
Model	NRMSE	NIS	NRMSE	NIS	NRMSE	NIS
Pro-NDF	**0.084**	0.464	0.140	0.675	0.080	0.400
LMGP	0.085	**0.413**	**0.112**	**0.540**	**0.074**	**0.365**
MF-DGP	0.095	0.885	0.197	1.056	0.453	2.936
SF-GP	0.111	0.889	0.327	2.114	6.423	3.43
FFNN	0.112	-	0.146	-	0.096	-
SMF	0.612	-	0.371	-	0.273	-

Table 2.2 Results on the analytic examples for different models: We test the performance of Pro-NDF against LMGP, SF-GP, MF-DGP, and two existing NN-based technologies for the Rational, Wing-weight and Borehole examples detailed in Table A.2. The training procedure for Pro-NDF, LMGP, FFNN and SMF is discussed in Section 2.3, Section 2.2.1, Appendix A.3.1 and, Appendix A.3.2 respectively. We report the NRMSE and NIS on unseen HF data. We provide MF-DGP with additional information that specifies the relative accuracy of the LF sources.

performance of Pro-NDF in NRMSE, while the SMF approach shows poor performance for all problems. We explain SMF's poor performance by noting that, as explained in Appendix A.3, hierarchical MF techniques such as SMF heavily rely on the knowledge of fidelity levels to process the data sources sequentially in the order of increasing accuracy. Since we assume in the problem setup that we only know which source has the highest fidelity and do not know the relative fidelity levels of the LF sources, the LF sources are ordered sub-optimally in the SMF approach which leads to a very poor prediction accuracy. The FFNN approach, by contrast, does not rely on the knowledge of fidelity levels and as such performs better than SMF. However, its performance lags behind that of LMGP and Pro-NDF because the architecture is not designed with MF problems in mind.

LMGP, which is considered as our gold standard for MF problems with small data, outperforms Pro-NDF in both NRMSE and NIS for the Wing-weight and Borehole problems, and in NIS for the Rational problem. The Rational problem is simultaneously the most data deficient and least complex of the problems examined in this work: as shown in Table A.2, there are 4 data sources with only one being especially inaccurate, the input and output are both $1D$, and there are only 5 training samples provided for the HF source. Pro-NDF and LMGP are well suited to tackle this problem as they both perform well for low-dimensional

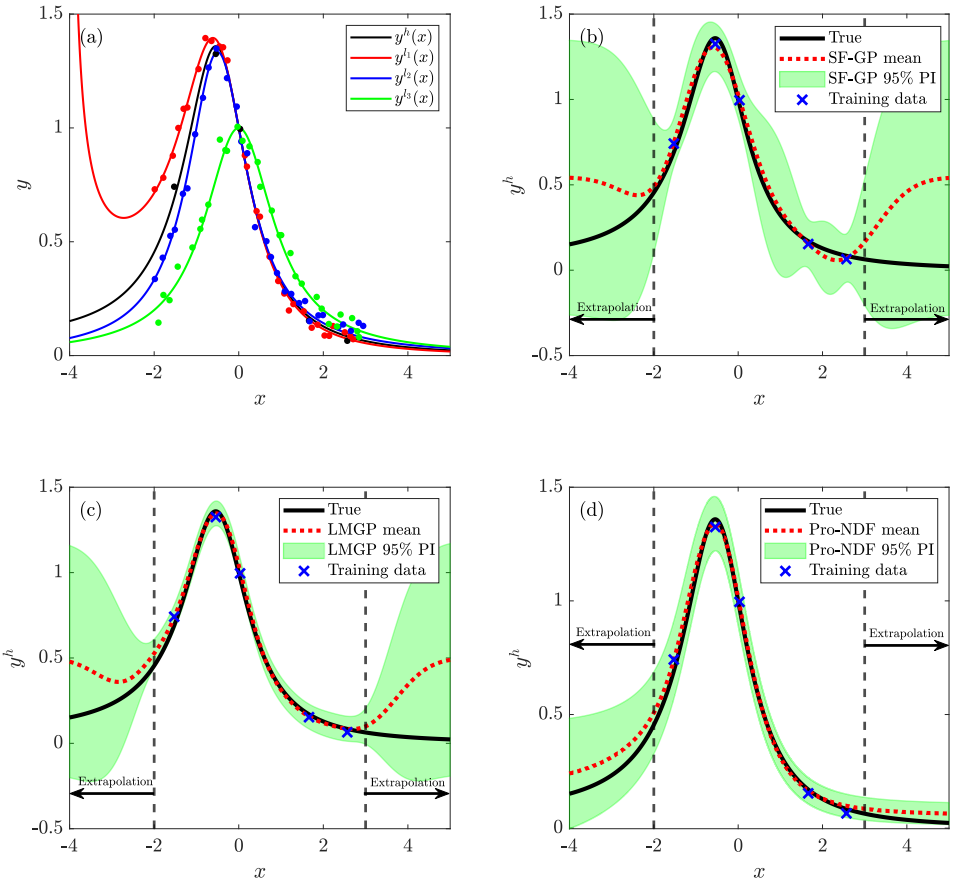

Figure 2.3 High-fidelity emulation on the Rational problem: a) Four data sources and the corresponding training data, b) SF-GP fit only to the HF data, c) LMGP fit to all available data and d) Pro-NDF fit to all available data. Pro-NDF and LMGP approaches clearly outperform SF-GP in terms of mean accuracy and give a narrower PI. They both produce similar results in terms of mean prediction in interpolation. However, LMGP has a narrower 95% PI and reverts to its mean in extrapolation.

problems with simple underlying functional forms and well-correlated sources, and as such they have similar performance. Figure 2.3 reveals that LMGP captures all of the training points in a narrower 95% PI compared to Pro-NDF which explains LMGP's lower NIS in Table 2.2. However, Pro-NDF shows a better performance for this problem in terms of mean prediction accuracy and it also has a higher degree of agreement with the true function in extrapolation while LMGP reverts to its mean. We therefore conclude that both methods perform on par on the Rational problem.

The learned fidelity manifold of Pro-NDF for the Rational problem is shown in Figure 2.4(a)

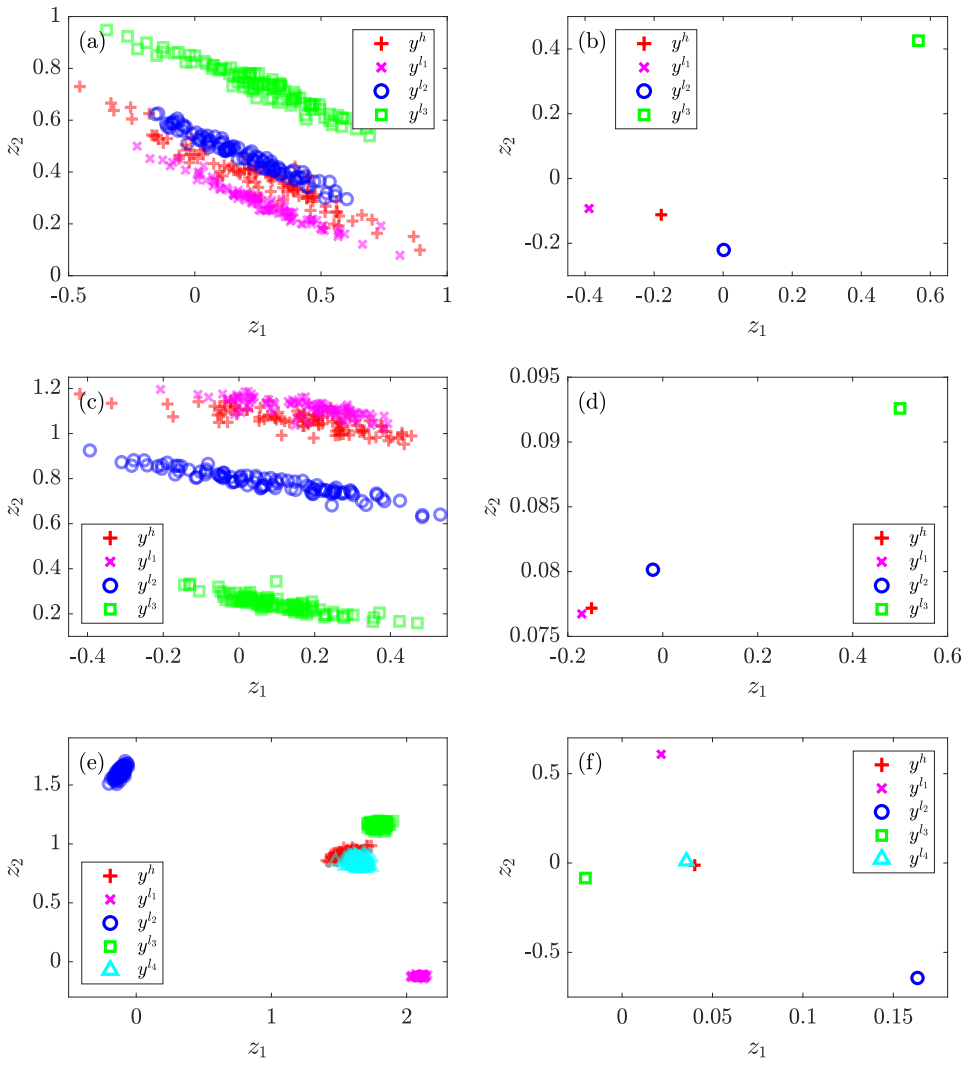

Figure 2.4 Pro-NDF and LMGP fidelity manifolds for the analytic problems: (a) Pro-NDF for Rational problem, (b) LMGP for Rational problem, (c) Pro-NDF for Wing-weight problem, (d) LMGP for Wing-weight problem, (e) Pro-NDF for Borehole problem, and (f) LMGP for Borehole problem.

which indicates that the network has inversely learned the true relationship between the data sources as y^{l_1} and y^{l_2} are encoded close to y^h while y^{l_3} is quite far from y^h. These relative distances are proportional to the accuracy of the LF sources with respect to the HF source (in the interpolation region) which are reported in Table A.2. The fidelity manifold also shows a high spread in the distributions of the realizations for individual sources which indicates either a poor fit to the data or a lack of training samples. In this case, we attribute this spread to lack of data since the performance in NIS and NRMSE is quite good.

The Wing-weight and Borehole problems are both high-dimensional problems with relatively complex underlying functional forms and small amounts of data. LMGP is very well suited to tackle this type of problem [33] because the number of its hyperparameters scales much better than NN-based approaches such as Pro-NDF . Accordingly, we observe that LMGP achieves lower NRMSE and NIS for both examples.

Comparing the performance of Pro-NDF across the two high-dimensional problems, we observe that it performs much better on the Borehole problem. Examining the fidelity manifold learned by Pro-NDF and LMGP for the Wing-weight problem, see Figure 2.4(c) and Figure 2.4(d), respectively, we see that both approaches accurately determine the relationship between the sources as they agree with the NRMSEs reported in Table A.2. Specifically, y^{l_1} is closer to y^h than y^{l_2}, which in turn is closer than y^{l_3}. Notably, both LMGP and Pro-NDF have the same relative ordering and positioning of the sources, i.e., (1) the mean position of all sources lies on an axis, and (2) y^{l_1} is in the opposite direction relative to y^h from y^{l_2} and y^{l_3}. This reinforces our earlier assertion in Section 2.3: the positions of the sources in the fidelity manifold learned by Pro-NDF reflect *correlations* between the data sources. However, the relative distances between the LF sources in the latent space found by LMGP more accurately represents the true relationships between the sources because the position for y^{l_3} is much more distant from y^h than encoded positions of the other sources.

In Figure 2.4(a) we observe a large spread in the realizations (i.e., the posterior distributions in the fidelity manifold are quite wide) which partially explains the poor[12] performance in this problem. We attribute this performance level to the relative accuracy of the data sources since only one source, y^{l_1}, is at all accurate with respect to y^h while the other LF sources are quite inaccurate. LMGP's performance is not inherently hampered by including poorly correlated sources in the data fusion problem [33] since its performance, upon successful optimization, is at worse on par with fitting separate GPs to each source. By contrast, since

[12]Poor with respect to LMGP. The performance of Pro-NDF is still much better than the other four approaches.

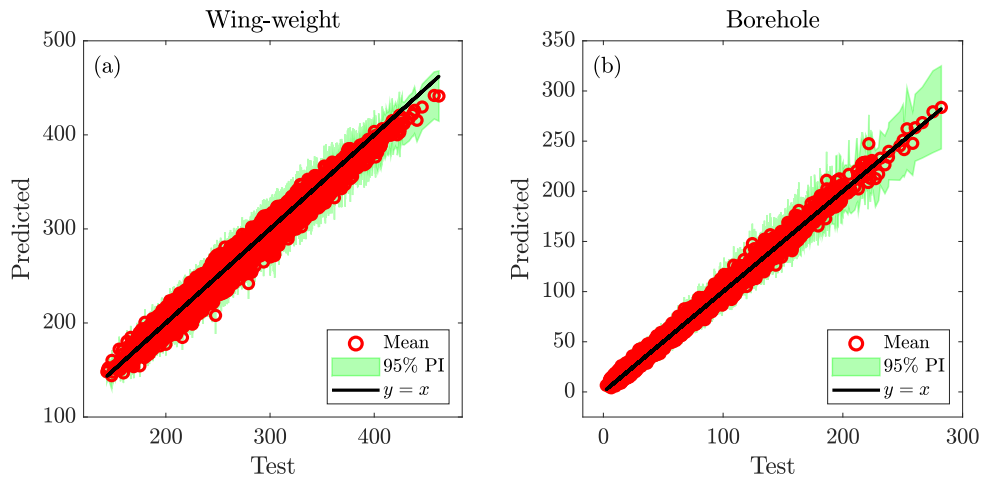

Figure 2.5 Predictions vs noisy test outputs: We compare the predictions of Pro-NDF on $10,000$ random HF samples for the two high-dimensional analytical problems. The noisy test data are within the 95% PIs which indicates that Pro-NDF is achieving a high performance in uncertainty quantification.

Pro-NDF's Block 3 is responsible for learning the relations between all sources and uses weight sharing, including especially inaccurate sources leads to relatively poor performance as shown in 2.4.1.

Lastly, we note that the performance of MF-DGPs, especially in uncertainty quantification (as measured by NIS), is much worse than both LMGP and Pro-NDF. This is an especially interesting result since we provide MF-DGP with the correct ordering of the data sources since it processes the data sequentially (e.g., for the Rational problem, MF-DGP knows that y^{l_2} is the most accurate LF source which is then followed by y^{l_1} and y^{l_3}). We attribute the deficiencies of MF-DGP to its sequential architecture which prevents the model from using all available data in emulating all the sources (e.g., as opposed to our approach, HF data does not directly contribute to learning any of the LF sources).

2.4.3 Real-World Problems

In this section, we validate our approach against LMGP, SF-GP, MF-DGP, and two existing NN-based technologies on two engineering applications which are detailed in Appendix A.2.

We again use NRMSE and NIS on HF test data as our performance metrics and examine the manifolds learned by Pro-NDF and LMGP. In both of these applications, the input space has categorical features (so Pro-NDF and LMGP each build two manifolds) and we do not know the underlying relationships between the data sources.

The results for each approach on each problem are summarized in Table 2.3 and demonstrate that the probabilistic approaches again significantly outperform the deterministic ones as well as MF-DGP which is provided with the additional information on the relative accuracy of each data source (this information is not provided to any of the other approaches). The FFNN approach performs nearly as well as LMGP and Pro-NDF in the DNS-ROM problem, but lags behind Pro-NDF and LMGP in the HOIP problem in terms of NRMSE. The SMF approach shows poor performance for both problems for the same reasons provided in Section 2.4.2. Notably, Pro-NDF outperforms LMGP for both problems in terms of both metrics which we partially explain by noting that there are much more data are available in these real-world problems compared to the analytical examples of Appendix A.1. Being an NN-based approach, Pro-NDF scales very well with additional data while the performance of LMGP has diminishing returns and eventually plateaus (recall that the latent map and

	DNS-ROM		HOIP	
Model	NRMSE	NIS	NRMSE	NIS
Pro-NDF	**0.101**	**0.520**	**0.470**	**1.855**
LMGP	0.105	0.706	0.473	2.510
MF-DGP	0.140	0.791	-	-
SF-GP	0.123	0.653	0.779	5.923
FFNN	0.113	-	0.580	-
SMF	0.130	-	0.663	-

Table 2.3 Results on the real-world examples for different models: We test the performance of Pro-NDF against LMGP, SF-GP, MF-DGP, and two existing NN-based technologies for the DNS-ROM and HOIP data sets detailed in Appendix A.2. We report the NRMSE and NIS on unseen HF data. We provide MF-DGP with additional information that specifies the relative accuracy of the LF sources. MF-DGP and vanilla GPs cannot handle categorical variables and hence for the HOIP example we do not apply MF-DGP and use LMGP (which can handle categorical inputs) instead of vanilla GPs for the SF-GP method.

kernel of LMGP are *not* tuned which contribute to this plateauing performance).

As shown in Figure 2.6(a-b), the fidelity manifolds learned by Pro-NDF and LMGP for the DNS-ROM problem are nearly analogous as the relative distances are quite similar. However, LMGP finds all sources to be on the diagonal axis while Pro-NDF learns a more nuanced relationship between the sources, which may contribute to its superior performance. We also observe that the spreads in the individual realizations for each point are fairly tight, which indicates that Pro-NDF is able to learn the relations between the sources reasonably well and, accordingly, provide good performance in terms of NRMSE and NIS. Like in Section 2.4.2, we show the HF predictions at test inputs in Figure 2.7 in order to facilitate the comprehension of the accuracy of Pro-NDF in these two problems.

The HOIP problem has three categorical inputs with 10, 3, and 16 levels and as such Pro-

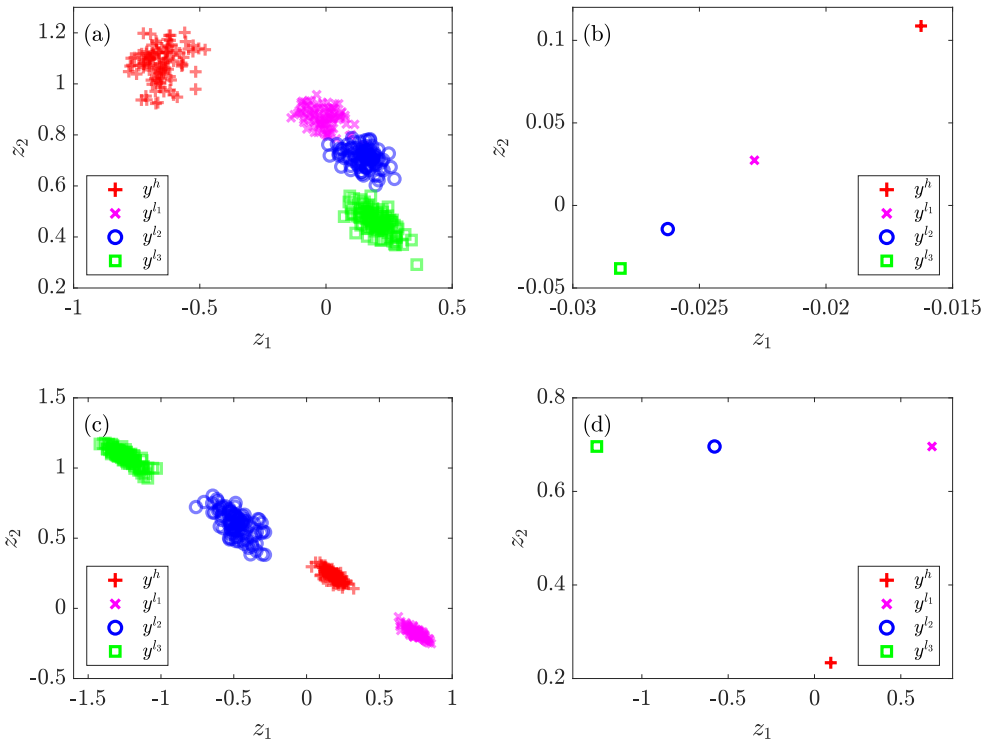

Figure 2.6 Pro-NDF and LMGP fidelity manifolds for the real-world problems: (a) Pro-NDF for DNS-ROM data set, (b) LMGP for DNS-ROM data set, (c) Pro-NDF for HOIP data set, (d) LMGP for HOIP data set.

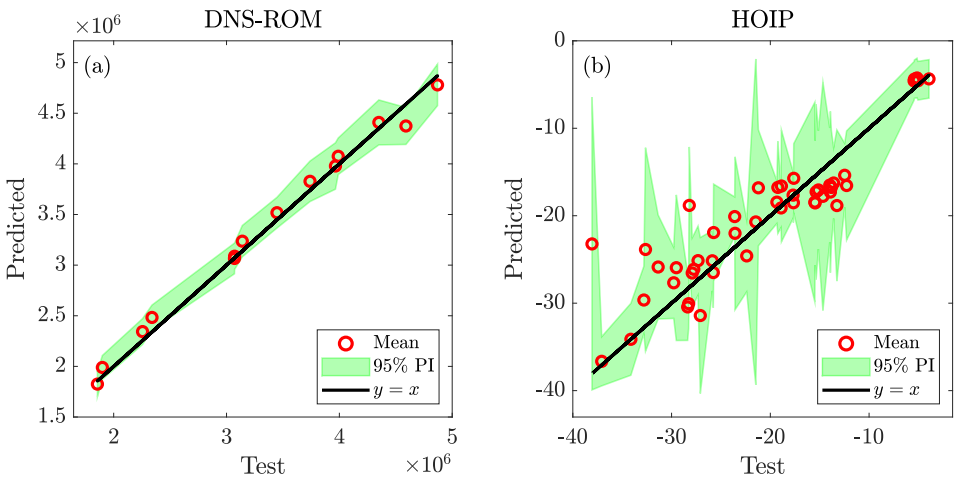

Figure 2.7 Predictions vs noisy test outputs: We compare the predictions of Pro-NDF on test data for the two high-dimensional engineering problems. The noisy test data are within the 95% PIs which indicates that Pro-NDF is achieving a high performance in uncertainty quantification.

NDF uses two separate latent transformations (one for the data source and the other for the three categorical variables) that correspond to Blocks 1 and 2 in Figure 2.1. The learned categorical manifolds for Pro-NDF and LMGP are shown in Figures 2.8 and 2.9 where the $\boldsymbol{z}^c$ is visualized four times as the combinations of the categorical variables are color-coded based on the levels of each of the three categorical variables and based on the average value of the output [13]. Since there are no numerical features, the combined inputs are $\boldsymbol{u} = [\boldsymbol{\zeta}(t^s), \boldsymbol{\zeta}(\boldsymbol{t}^c)]$ and $\boldsymbol{\nu} = [\boldsymbol{z}^s, \boldsymbol{z}^c]$ are the inputs to Block 3 of Pro-NDF . Recall that we only use a BNN in Block 1 and as such we show only one realization for the manifold for Pro-NDF that encodes the categorical variables.

Pro-NDF outperforms LMGP in terms of NRMSE by a small margin and NIS by a significant margin for this problem which we attribute to the size of the data sets. Pro-NDF is able to leverage these additional data much more readily than LMGP which only uses simple mapping functions to handle categorical variables t^s and $\boldsymbol{t}^c$. Pro-NDF also finds fairly tight spreads in the probabilistic fidelity manifold, see Figure 2.6(c); indicating that it has high certainty in its outputs and that we should expect good performance. We note that all sources are found to be roughly on one axis and roughly spaced evenly from each other, which

[13] This average is obtained using the entire data set including both the training and test data.

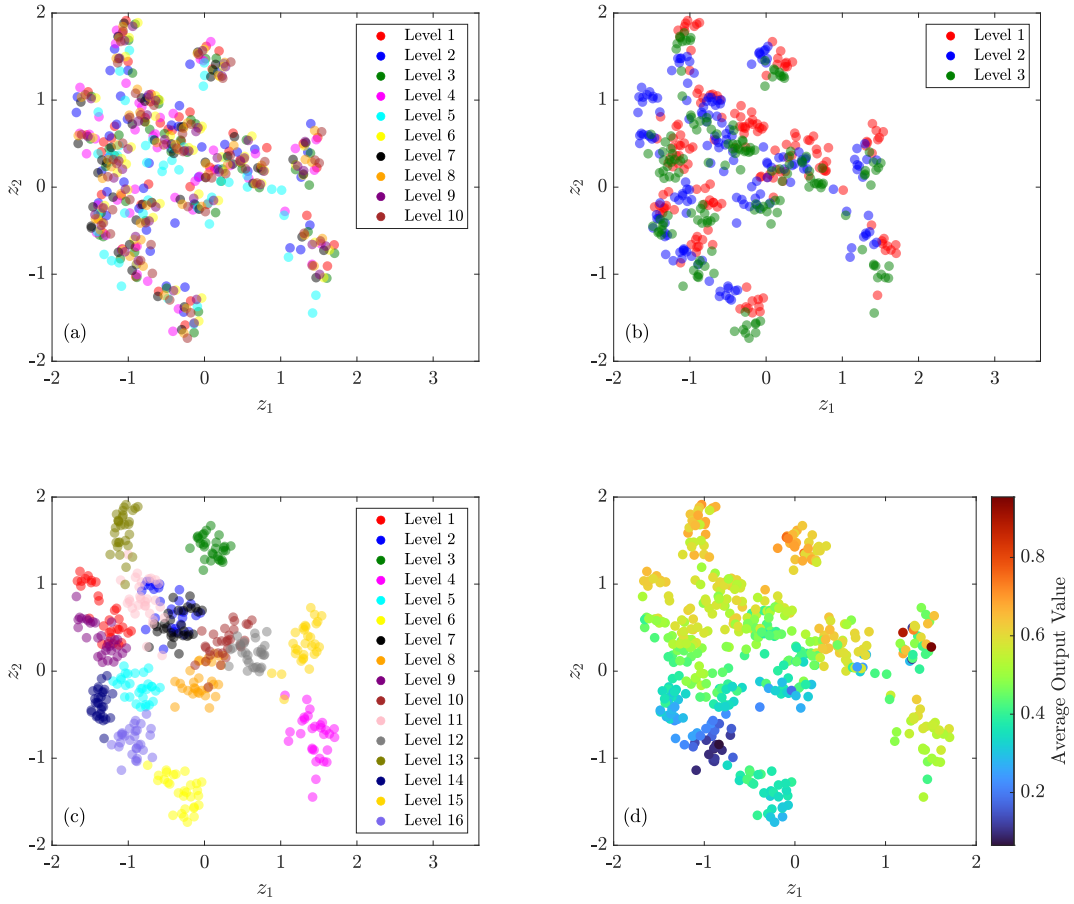

Figure 2.8 Pro-NDF categorical manifold for the HOIP problem: The combination of the categorical variables' levels are color-coded based on: (a) the levels of t_1^c, (b) the levels of t_2^c, (c) the levels of t_3^c, (d) the average output value.

may indicate that Pro-NDF has failed to learn the more nuanced relationships between the sources. Equally likely, however, is that the relationships between the sources are simple enough to be represented in this way; since we do not know the underlying functional forms for this problem, we cannot give a definitive answer.

We can also glean some information about the relationships between the categorical variable levels and their impact on the output by examining the corresponding manifolds in Figures 2.8 and 2.9. Figure 2.8(c) shows that Pro-NDF finds distinct clusters for all 16 levels of t_3^c which indicates that distinguishing between the levels of t_3^c is important to learning the output. Similarly, the levels of t_2^c are distinguishable in Figure 2.8(b) as t_2^c affects the

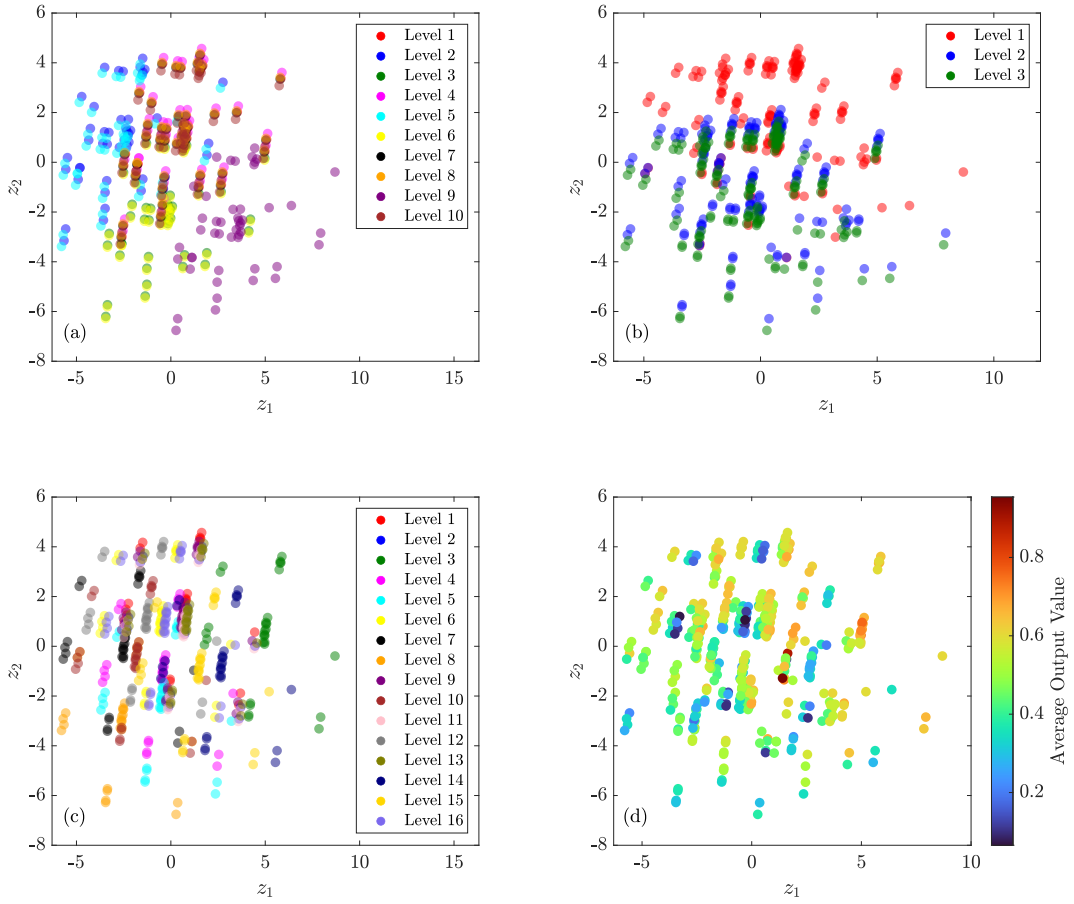

Figure 2.9 LMGP categorical manifold for the HOIP problem: The combination of the categorical variables' levels are color-coded based on: (a) the levels of t_1^c, (b) the levels of t_2^c, (c) the levels of t_3^c, (d) the average output value.

response value. By contrast, Figure 2.8(a) shows no apparent trend between the 10 levels of t_1^c which implies that t_1^c has little effect on the output as Pro-NDF does not learn to distinguish the levels from each other. By contrast, the manifold found by LMGP, shown in Figure 2.9 shows much less distinct clustering for each of the three categorical variables, which may help explain why it achieves a lower NIS than Pro-NDF . Finally, we examine whether the latent positions for the categorical combinations are influenced by the average output value in Figure 2.8(d) and Figure 2.9(d). The manifold for Pro-NDF shows a clear trend of the average output value increasing as the latent points move from the bottom-left of the space to the top-right, while for LMGP there is no obvious trend. Based on these

manifolds, Pro-NDF shows superior ability to discern relationships between the categorical combinations and between levels of categorical variables.

Lastly, we observe MF-DGP provides the worst performance in the DNS-ROM example even though it is provided with the correct ordering of the data sources. We believe MF-DGP does not perform well because it sequentially stacks a set of GPs that (1) prevents bi-directional information flow between any two sources where the data from source i informs the model about source j only if source j is believed to be more accurate than it, and (2) if two sources are not immediately related by two sequential GPs, their relations is only indirectly learnt via intermediate GPs.

2.4.4 Convergence Study

In this section, we perform a convergence study to assess the performance of Pro-NDF against three other methods on three analytical examples while varying the sizes of the data sets. We compare our approach to LMGP, MF-DGP, and multi-fidelity neural networks (MFNN) [36]. All the examples in this subsection are bi-fidelity since MFNN can only fuse two data sources (for this reason MFNN is not used in Section 2.4.2 and Section 2.4.3). The three problems studied here are listed in Table A.2 and include the Polynomial function of [62] and the adapted versions of the Wing-weight and Borehole examples where we only use two data sources ($y^h(\boldsymbol{x})$ and $y^{l_2}(\boldsymbol{x})$ for the Wing-weight, and $y^h(\boldsymbol{x})$ and $y^{l_3}(\boldsymbol{x})$ for the Borehole).

We show the results of the convergence study in Figure 2.10 in terms of NRMSE and NIS for different data set sizes. The size of the initial data sets (at $k = 1$) for the three problems are $n_1^h = 5$ and $n_1^l = 20$. The sizes of the subsequent data sets are increased via a multiplicative factor k, i.e., $n_k^h = k \times n_1^h$ and $n_k^l = k \times n_1^l$. The results show that all methods, including Pro-NDF, provide higher accuracy as the number of samples for both sources increase. We also observe that for high-dimensional problems (Wing-weight and Borehole), the GP-based

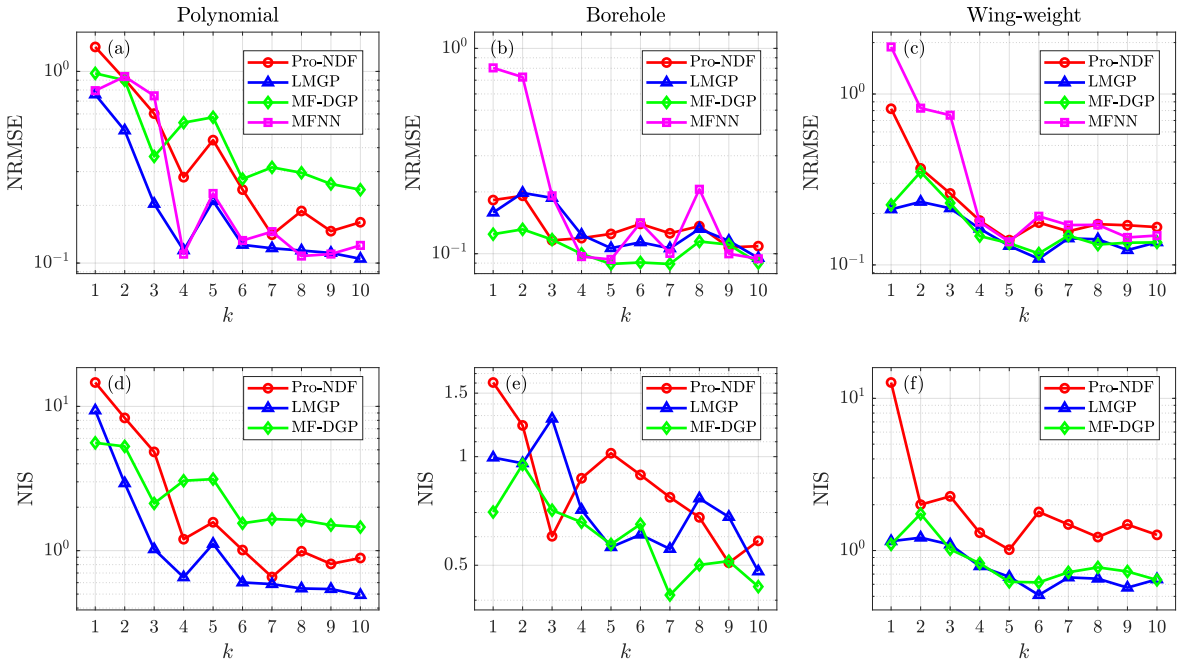

Figure 2.10 Convergence study: We compare four data fusion methods in terms of NRMSE and NIS across three examples as the sizes of the data sets increase. The Polynomial (a,d), Borehole (b,e) and Wing-weight (c,f) problems are detailed in Table A.2. The size of the initial data sets (at $k = 1$) for the three problems are $n_1^h = 5$ and $n_1^l = 20$. These sizes increase as $n_k^h = k \times n_1^h$ and $n_k^l = k \times n_1^l$.

approaches are more accurate when data sets are small (e.g., when $k \leq 4$).

In these bi-fidelity examples, both MFNN and MF-DGP naturally leverage the hierarchy between the LF and HF sources since both of them are structured such that the information from the LF source is directly propagated to the part of the model that surrogates the HF source. However, LMGP and Pro-NDF do not leverage this information in that they both aim to emulate both sources as accurately as possible using the cross-correlation between the LF and HF data sets. As we demonstrate in Section 2.4.2 and Section 2.4.3 such hierarchical approaches provide poor performance in applications where there are more than two data sources.

Despite not being designed for bi-fidelity problems, Pro-NDF achieves a comparable performance to the other state-of-the-art methods in these examples. Rather, Pro-NDF is better suited for MF problems with more than two sources, which we demonstrate by comparing the

results in Figure 2.10 with the ones from previous subsections. For instance, in Figure 2.10 we can see that MF-DGP is able to outperform Pro-NDF in some data sets for the Borehole and Wing-weight examples. However, when using all the available data (not just two sources of data), we observe in Section 2.4.2 and Section 2.4.3 that Pro-NDF significantly outperforms MF-DGP for all the examples (in fact, in both Borehole and Wing-weight examples, the performance of MF-DGP drops once it is provided with LF data from two additional sources, compare Figure 2.10 to Table A.2).

2.5 Conclusions

In this work, we introduce Pro-NDF for data fusion under uncertainty. Pro-NDF is based on a multi-block NN where each block is designed to take on specific tasks for MF modeling problems that arise in typical engineering applications. One of these blocks is probabilistic whose visualizable output can be used to detect LF sources with large model form errors. The final output of Pro-NDF is also probabilistic which enables to not only quantify aleatoric uncertainties, but also leverage strictly proper scoring rules during training.

We validate each of the key components of Pro-NDF by performing an ablation study on an analytic and a real-world example. We also demonstrate that on multi-source problems Pro-NDF outperforms other NN-based data fusion approaches by a large margin. Moreover, Pro-NDF performs on par to LMGP in low-dimensional cases with small data sets and slightly lags behind LMGP (a competing GP-based approach) in high-dimensional examples with very small data sets. However, as the size of the training data increases, Pro-NDF scales better than LMGP and provides smaller errors. We obeserve based on the reported NIS that Pro-NDF performs comparably to LMGP at avoiding overconfidence, which we attribute to our novel loss function. In these studies, we test the performance on unseen HF data but note that Pro-NDF builds an MF emulator that probabilistically surrogates all the

data sources simultaneously.

Our convergence study indicates (1) the performance of Pro-NDF, similar to other MF modeling techniques, improves as the data set sizes increase, and (2) the advantages of our approach are more pronounced in problems with more than two data sources as we can use data on any source to better learn another source.

A particularly useful output of Pro-NDF is its learnt fidelity manifold which encodes source-wise similarities/discrepancies. While the learnt distances in this manifold do not directly link correlation between the sources, we observe that the fidelity manifold of Pro-NDF and LMGP look quite similar in our studies. Since the fidelity manifold of LMGP is embedded in its kernel and hence indicates the correlations, we believe the fidelity of Pro-NDF also estimates a scaled version of correlation. An added benefit of Pro-NDF's fidelity manifold is that it is probabilistic where wide distributions can indicate if Pro-NDF is able to learn the relation between the data sources. Reducing this uncertainty via domain knowledge (especially qualitative information in engineering applications) is a future direction that we plan to investigate.

The performance of any data fusion approach (including ours) can drop if there are one or more very inaccurate LF sources. With Pro-NDF , the learned fidelity manifold can be used to identify-discard such sources and then retrain Pro-NDF anew. This process can be repeated until all LF sources are encoded close to the HF source in the fidelity manifold. This iterative approach is, however, quite inefficient so we plan to develop an automated mechanism that perhaps leverages the fidelity manifold to adjust the loss function and, in turn, prevent Pro-NDF from learning the highly inaccurate LF sources.

Finally, we stress the fact that the fidelity manifold currently provided by Pro-NDF is only a function of the source indicator, thereby providing a global correlation measure between the sources. However, for future work, we aim to extend Pro-NDF to include a fidelity

manifold that is also dependent on the inputs. This extended approach would enable the learning of a local correlation measure, conditioned on the inputs, thus allowing for the detection of regions in the feature space where the correlation between sources is either high or limited. Another possible direction involves changing the BNN blocks in the architecture to another probabilistic or stochastic model. For example, dropout regularization acts as an approximation of Bayesian inference and has a similar interpretation as an ensemble [63] while providing better computational tractability than BNNs.

Chapter 3

Data-Driven Calibration of Multi-fidelity Multi-scale Fracture Models Via Latent Map Gaussian Process

3.1 Motivation

Multiscale models are increasingly employed to quantify the effects of manufacturing-induced microscopic defects on the performance of macroscopic components. In such models, a microstructure or a representative volume element (RVE) is associated with each integration point (IP) of the discretized macrostructure. Traditional multiscale simulations use the finite element method (FEM) to solve the nonlinear equilibrium equations at both scales where macroscopic deformation gradients $\mathbf{F}^{\mathrm{M}}$ and RVE effective stress $\boldsymbol{\sigma}^{\mathrm{M}}_{\mathrm{FEM}}$ are exchanged between the two scales at each iteration, see Figure 3.1(a). A major challenge associated with such

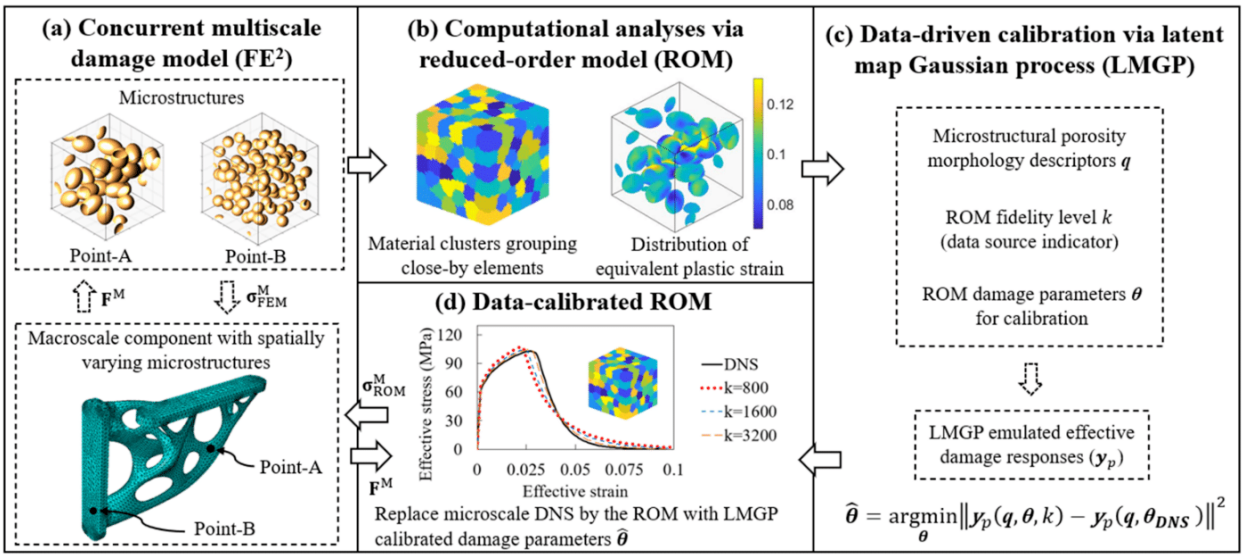

Figure 3.1 Proposed data-driven framework for multiscale damage modeling: LMGP creates a multi-fidelity emulator for the ROMs and DNS. It is then used in an inverse optimization to determine the damage parameters that must be used in ROMs such that they approximate DNS as closely as possible conditioned on the microstructure. Upon this calibration, a multiscale simulation is run where ROMs are used at the microscale.

nested simulations is the computational expenses which prohibitively increase in the presence of nonlinear microscale deformations that involve damage. Reducing these costs holds the key to understanding the relation between microscopic defects and components' fracture behavior and, in turn, guiding the "design for fracture" process. To this end, we propose a data-driven framework that has two major components: (1) a mechanistic reduced-order model (ROM) with an adjustable degree of fidelity, and (2) a multi-fidelity modeling and calibration scheme based on latent map Gaussian processes (LMGPs) [40, 33]. Integration of these two components enables us to build calibrated multi-fidelity ROMs that can simulate the damage behavior of multiscale materials with spatially varying microstructures.

The rest of this chapter is organized as follows. In Section 3.2, we review existing works on reduced-order modeling and discuss the research gaps that we aim to address. The overview and technical details of our approach are provided in Section 3.3 and 3.4, respectively. We evaluate the performance of our approach in Section 3.5 and draw conclusions in Section 3.6.

3.2 Background on reduced-order modeling

Mechanistic ROMs are increasingly employed to accelerate nonlinear materials modeling by using a combination of methods from linear algebra and machine learning that result in reducing the number of unknown variables that characterize, e.g., microstructural strain and stress fields. Transformation field analysis (TFA) and its successor nonuniform transformation field analysis (NTFA) are two of the earliest ROMs [64, 65, 66]. These two methods approximate plastic strain as either piecewise constants or spatially varying orthonormal eigenstrains which are pre-selected in an offline stage. These eigenstrains evolve in the online stage based on pre-defined analytical functions that involve thermodynamic forces and potentials.

Clustering-based ROMs are recent techniques that decompose microstructure domains into a set of clusters whose interactions and deformations are modeled. For instance, the self-consistent analysis (SCA) [67] lumps material points with similar elastic responses and then quantifies cluster-to-cluster interactions by the incremental Lippmann-Schwinger equation. Finite element-based cluster analysis [68] approximates the microstructural effective responses by following the cluster minimum complementary energy principle. Deflated clustering analysis (DCA) [69] agglomerates close-by material integration points (IPs) into clusters and the cluster-wise quantities of interests are computed in a multi-grid fashion where unknown variables are projected back and forth between different meshes. In this work, we use cluster-based ROMs as they provide higher efficiency and versatility compared to other methods such as TFA.

Successful application of any ROM depends on two primary factors: (1) the coarsening degree (e.g., the chosen number of clusters) which makes a tradeoff between fidelity level and computational costs, and (2) the calibrated material properties. Both of these factors depend on the microstructure as well as the properties of interests. For example, accurate

prediction of the damage behavior requires different damage parameters and a number of clusters for the two microstructures in Figure 3.1(a). In particular, given a desired level of accuracy with respect to high-fidelity direct numerical simulations (DNS), the analysis of the more complex microstructure in Figure 3.1(b) generally requires more clusters (i.e., less coarsening or data reduction).

Regarding the second requirement of the successful application of ROMs, we note that accurate prediction of damage behavior necessitates the calibration of material properties to account for the diffusive stress and strain fields of any ROM. The diffusion typically depends on the microstructure topology, and it unrealistically increases the tolerance of the material system to localized phenomena. The superficial increase of material strength upon clustering, therefore, must be counteracted in ROM to ensure solution accuracy. We clarify that, in this paper, we use the word 'diffusion' to exclusively refer to the artificially strengthening of material clusters in ROMs (which aim to capture the homogenized behavior of the material encompassed in the cluster) and not the transfer of matter by diffusion.

While calibrating material properties plays a vital role in ensuring that ROMs can be reliably used in multiscale simulations, there is still a lack of systematic approaches that dispense with manual calibration which is time-consuming and suboptimal. As explained in the next sections, our contribution is to develop a data-driven framework to automate the calibration process of ROMs with any fidelity level as a function of material morphology.

3.3 Proposed Framework

Our framework relies on two primary components for damage modeling in multiscale metals with porosity: a novel cluster-based ROM and LMGP-based calibration which are detailed in Section 3.4.3 and Section 2.2.1, respectively. We provide an overall description of these

components in this section.

The ROM surrogates DNS and estimates the stress field in a microstructure under arbitrary displacement boundary conditions that may result in plasticity and damage. The fidelity of the ROM is determined by the user-defined parameter k which indicates the number of clusters and balances costs and accuracy.

As argued in Section 3.2, the material properties that must be used in ROM should be different than the true values that are used in DNS, i.e., the ROM requires calibration. This difference depends on both the microstructure complexity and, more importantly, k. Hence, we use a data-driven approach that relies on emulation via an LMGP to calibrate the material properties for ROMs. In particular, we use the trained LMGP emulates ROM and DNS to answer the following question:

Given k and one microstructure, what damage parameters should be used in the ROM such that it predicts the same fracture response as DNS which uses the true damage parameters?

We answer the above question by solving an inverse optimization problem whose objective function relies on LMGP, see Figure 3.1(c). To make the optimization problem tractable, we make two mild assumptions. Firstly, we consider a small set of integer values for k, i.e., we assume $k = 800$,1600, or 3200 but more values can be used within our framework. As shown in Section 3.4.3, all these values are much smaller than the number of elements in a typical mesh used in DNS and hence result in massive data reduction or coarsening. Secondly, the very high dimensional morphology of microstructures can be represented with a reduced set of quantitative descriptors that in our case characterize the geometry and spatial distribution of the pores.

We note that the clustering-based ROMs are new methods developed in recent years. Even though ROMs dramatically improve simulation efficiency, the number of clusters in ROMs is currently chosen in an ad-hoc manner and there still lacks theoretical proof on the criteria

to choose the number of clusters for arbitrary material systems. This is because material systems can be very different in local morphology, material composition/property, defect types/distributions, etc. More complex material systems generally require higher fidelity models and therefore more clusters. The goal of our work is to replace this manual approach for selecting the number of clusters with an automated method. In this work, we start with 800 clusters and then doubled this number twice to get 1600 and 3200 clusters. We intentionally did not study lower/higher clusters because we aim to develop a rather general calibration scheme that is not sensitive to the chosen cluster numbers.

To build the LMGP, we generate the training samples by the design of experiments (DoE) where the inputs are microstructural descriptors and calibration parameters that control the damage behavior. For sample i, we first use a reconstruction algorithm to build the microstructure corresponding to the i^{th}set of descriptors. Then, we calculate the fracture response of the i^{th} microstructure via a simulator (i.e., DNS or one of the ROMs) while using the i^{th} set of damage parameters. When obtaining the responses, we select the simulator based on its computational costs, i.e., the frequency of using a simulator is inversely proportional to its costs (e.g., we employ an ROM with small k much more than DNS or an ROM with large k).

It is noted that the optimization problem uses LMGP rather than a traditional Gaussian process (GP) since we view the data source indicator as a categorical input rather than a quantitative one, see Figure 3.1(c). This choice is justified since alternating the data source (e.g., DNS vs. ROM with $k = 800$ vs. ROM with $k = 3200$) encodes the diffusive nature of strain-stress fields which cannot be readily characterized with quantitative inputs. Hence, our treatment of data source motivates the use of LMGP and greatly simplifies the emulation as it eliminates the manual conversion of the source label to a quantitative variable.

Once LMGP is built, we are ready to run a multiscale simulation where ROMs are used at the microscale instead of DNS, see Figure 3.1(d). We first assign spatially varying mi-

crostructures to the IPs on the macro-component. Then, based on the complexity of the microstructures and any prior knowledge (if available) on the macro-locations where excessive deformations can occur (e.g., near sharp corners), we choose the k values for ROM. Next, we use the trained LMGP to assign the damage parameters that must be used at i^th macro-IP given the k and microstructure assigned to it. Upon this assignment, we conduct the multiscale simulation to find the performance of the macro-component while considering microstructural porosity.

3.4 Technical details

We first provide the details on damage modeling with our ROM in Section 3.4.1 through 3.4.3. Then, we explain our optimization-based calibration algorithm in Section 3.4.4. For details on the training process of LMGPs, we refer the reader to Section 2.2.1.

3.4.1 Stabilized micro-damage model

Damage includes strain-softening which causes convergence issues in implicit time integration schemes. To address this issue, we use a stabilized damage model [70] to simulate microstructural effective responses during fracture progression. This model decouples damage evolution from elasto-plasticity by introducing three reference RVEs that share state variables with the original damaged RVE. By tracing the elasto-plasticity in one of the referenced RVEs via a classic implicit scheme, the effective fracture stress and states can be mapped to the damaged RVE.

The homogenized damage stress in an arbitrary RVE can be written as:

$$\mathbf{S}_{\mathrm{M}}^{\mathrm{d}} = \mathbb{C}_{\mathrm{M}}^{\mathrm{d}} : \mathbf{E}_{\mathrm{M}}^{\mathrm{el}} = \mathbb{C}_{\mathrm{M}}^{\mathrm{d}} : \left(\mathbf{E}_{\mathrm{M}} - \mathbf{E}_{\mathrm{M}}^{\mathrm{pl}}\right) \tag{3.1}$$

where $\mathbf{S}_{\mathrm{M}}^{\mathrm{d}}$ represents the effective damage stress, $\mathbb{C}_{\mathrm{M}}^{\mathrm{d}}$ is the homogenized damaged tangent modulus matrix, $\mathbf{E}_{\mathrm{M}}$, $\mathbf{E}_{\mathrm{M}}^{\mathrm{el}}$ and $\mathbf{E}_{\mathrm{M}}^{\mathrm{pl}}$ are the RVE effective strain, elastic strain, and plastic strain, respectively. The subscript M indicates that the variable is a macroscopic quantity. The symbol ":" represents the double dot product that contracts a pair of repeated indices.

The first reference RVE shares the same elasto-plastic deformation as the original RVE but is not damaged. Its effective stress is therefore computed as

$$\mathbf{S}_{\mathrm{M}}^{1} = \mathbb{C}_{\mathrm{M}}^{\mathrm{el}} : \mathbf{E}_{\mathrm{M}}^{\mathrm{el}} = \mathbb{C}_{\mathrm{M}}^{\mathrm{el}} : \left(\mathbf{E}_{\mathrm{M}} - \mathbf{E}_{\mathrm{M}}^{\mathrm{pl}}\right) \tag{3.2}$$

where $\mathbf{S}_{\mathrm{M}}^{1}$ and $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$ represent the homogenized stress and the undamaged elastic modulus, respectively (superscript 1 refers to the first referenced RVE). By combining Equation (3.1) and 3.2, we can express the referenced stress as

$$\mathbf{S}_{\mathrm{M}}^{1} = \mathbb{C}_{\mathrm{M}}^{\mathrm{el}} : \left(\mathbb{C}_{\mathrm{M}}^{\mathrm{d}}\right)^{-1} : \mathbf{S}_{\mathrm{M}}^{\mathrm{d}} \tag{3.3}$$

The second reference RVE has the same effective stress ($\mathbf{S}_{\mathrm{M}}^{2} = \mathbf{S}_{\mathrm{M}}^{1}$) and material property as the first RVE but is assumed to deform elastically.Thus, its effective elastic strain $\mathbf{E}_{\mathrm{M}}^{\mathrm{el}}$ is

$$\mathbf{E}_{\mathrm{M}}^{\mathrm{el}} = \left(\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}\right)^{-1} : \mathbf{S}_{\mathrm{M}}^{1} = \left(\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}\right)^{-1} : \mathbf{S}_{\mathrm{M}}^{2} \tag{3.4}$$

The effective stress and strain of the second reference RVE are equivalently expressed as the volume average of its microscale stress and strain as

$$\mathbf{S}_{\mathrm{M}}^{2} = \frac{1}{|\Omega|} \int_{\Omega} \mathbf{S}_{\mathrm{m}}^{2} d\Omega \tag{3.5}$$

$$\mathbf{E}_{\mathrm{M}}^{\mathrm{el}} = \frac{1}{|\Omega|} \int_{\Omega} \mathbf{E}_{\mathrm{m2}}^{\mathrm{el}} d\Omega \tag{3.6}$$

where $|\Omega|$ is the RVE volume, the subscript m indicates that the variable is a microscopic

quantity, and the microscale stress $\mathbf{S}_{\mathrm{M}}^{2}$ is proportional to the microscale elastic strain $\mathbf{E}_{\mathrm{m2}}^{\mathrm{el}}$ at any microscopic point by the elastic modulus $\mathbb{C}^{\mathrm{el}}$ via

$$\mathbf{S}_{\mathrm{m}}^{2} = \mathbb{C}^{\mathrm{el}} : \mathbf{E}_{\mathrm{m2}}^{\mathrm{el}} \tag{3.7}$$

The third reference RVE has the same elastic strain as the second one $\left(\mathbf{E}_{\mathrm{m3}}^{\mathrm{el}} = \mathbf{E}_{\mathrm{m2}}^{\mathrm{el}}\right)$ but its modulus is assumed to be identical to the original fractured RVE as

$$\mathbf{S}_{\mathrm{m}}^{3} = \mathbb{C}_{\mathrm{m}}^{\mathrm{d}} : \mathbf{E}_{\mathrm{m3}}^{\mathrm{el}} \tag{3.8}$$

$$\mathbb{C}_{\mathrm{m}}^{\mathrm{d}} = (1 - D_m)\mathbb{C}^{\mathrm{el}} \tag{3.9}$$

where $\mathbb{C}_{\mathrm{m}}^{\mathrm{d}}$ is the microscale damaged tangent modulus and D_m is the damage parameter at a microscopic IP. The value of D_m is determined by the plastic strain states in the first reference RVE

$$D_{\mathrm{m}}\left(\bar{E}^{\mathrm{pl}}; \alpha, \bar{E}^{\mathrm{cr}}\right) = 1 - \frac{\bar{E}^{\mathrm{cr}}}{\bar{E}_{\mathrm{m1}}^{\mathrm{pl}}} \exp\left(-\alpha\left(\bar{E}_{\mathrm{m1}}^{\mathrm{pl}} - \bar{E}^{\mathrm{cr}}\right)\right) \tag{3.10}$$

where $\bar{E}^{\mathrm{pl}}$ is the equivalent plastic strain, $\bar{E}_{\mathrm{m1}}^{\mathrm{pl}}$ is the equivalent plastic strain at a microscale IP in the first referenced RVE, and $\bar{E}^{\mathrm{cr}}$ is the critical plastic strain. α is the damage evolutionary rate parameter and a larger value of α results in faster material degradations and rapid effective stress drop amid softening. We note that local damage is initiated ($D_m = 0$) when the effective plastic strain equals the critical strain $\left(\bar{E}_{\mathrm{m1}}^{\mathrm{pl}} = \bar{E}^{\mathrm{cr}}\right)$ and damage reaches total rupture ($D_m = 1$) when the effective plastic strain is much larger than the critical plastic strain.

The effective damaged stress of the original RVE is assumed to be equal to the homogenized stress of the third reference RVE and is calculated as

$$\mathbf{S}_{\mathrm{M}}^{\mathrm{d}} = \mathbf{S}_{\mathrm{M}}^{3} = \frac{1}{|\Omega|}\int_{\Omega} \mathbf{S}_{\mathrm{m}}^{3} d\Omega \tag{3.11}$$

For the multiscale damage analysis in Section 3.5.4, the macroscale damage parameter is computed as the ratio of the norms of effective stress tensors of the original and the first reference RVE as

$$D_{\mathrm{M}} = 1 - \frac{\left\|\mathbf{S}_{\mathrm{M}}^{\mathrm{d}} : \mathbf{S}_{\mathrm{M}}^{1}\right\|}{\|\mathbf{S}_{\mathrm{M}}^{1} : \mathbf{S}_{\mathrm{M}}^{1}\|} \tag{3.12}$$

where D_M is the homogenized damage parameter representing the fracture status of a macroscale IP (and its associated RVE) on a macroscale component.

3.4.2 Condensation method

When using the stabilized microdamage model of Section 3.4.1 in a multiscale simulation, the effective elastic tangent moduli $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$ elastic tangent moduli is needed at each macroscopic IP, see Equation (3.2). Since we assign spatially varying RVEs with complex morphologies to macro-IPs, $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$ M needs to be computed via variational principles for each RVE [71]. This numerical procedure is needed since the constitutive laws of the RVEs are not available in closed form.

As variational calculations are expensive, we employ the condensation method [72] to compute the effective tangent modulus of an RVE. The condensation method starts by partitioning the microstructural system of equations as

$$\begin{bmatrix} \mathbf{K_{pp}} & \mathbf{K_{pf}} \\ \mathbf{K_{fp}} & \mathbf{K_{ff}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{u_p} \\ \delta\mathbf{u_f} \end{bmatrix} = \begin{bmatrix} \delta\mathbf{f_p} \\ \mathbf{0} \end{bmatrix} \tag{3.13}$$

where $\delta\mathbf{u_p}$ and $\delta\mathbf{u_f}$ represent the displacement variations at the prescribed and free nodes, respectively, in an RVE where the indices $\mathbf{p}$ and $\mathbf{f}$ represent the prescribed and free degrees of freedom, and $\delta\mathbf{f_p}$ is the external force on the nodes with prescribed forces. $\mathbf{K_{pp}}$, $\mathbf{K_{pf}}$, $\mathbf{K_{fp}}$, and $\mathbf{K_{ff}}$ are the corresponding partitions of the RVE's stiffness matrix. Eliminating $\delta\mathbf{u_f}$ from Equation (3.13) leads to a reduced system, with a reduced stiffness $\mathbf{K_r}$ which directly

relates the variations of the prescribed displacements with nodal forces

$$\mathbf{K_r}\delta\mathbf{u_p} = \delta\mathbf{f_p} \tag{3.14}$$

$$\mathbf{K_r} = \mathbf{K_{pp}} - \mathbf{K_{pf}}\left(\mathbf{K_{ff}}\right)^{-1}\mathbf{K_{fp}} \tag{3.15}$$

To transform $\mathbf{K_r}$ to the tangent modulus that relates variations of stress and strain, we substitute Equation (3.14) into the variational form of the macroscopic stress

$$\mathbf{S}_{\mathrm{M}}(\mathbf{X}) = \frac{1}{|\Omega_{0\mathrm{m}}|}\int_{\Gamma_{0\mathrm{m}}} \bar{\mathbf{t}}_{\mathrm{m}} \otimes (\mathbf{x} - \mathbf{x}_0)\, d\Gamma \tag{3.16}$$

where $\mathbf{x}$ and $\mathbf{x}_0$ are the microscale IPs at the deformed and original configurations, $\mathbf{S}_{\mathrm{M}}$ is the macroscale stress at the macroscopic IP $\mathbf{X}$, $\bar{\mathbf{t}}_{\mathrm{m}}$ is the microscale surface traction, $\Gamma_{0\mathrm{m}}$ is the RVE boundary, and $\otimes$ denotes the tensor product between $\bar{\mathbf{t}}_{\mathrm{m}}$ and the position vector $(\mathbf{x} - \mathbf{x}_0)$. Upon some algebraic modifications, the homogenized tangent (elastic) modulus matrix of an RVE can be obtained as

$$\mathbb{C}_{\mathrm{M}}^{\mathrm{el}} = \frac{1}{|\Omega_{0\mathrm{m}}|}\left[(\mathbf{x} - \mathbf{x}_0) \otimes \mathbf{K}_{\mathrm{r}} \otimes (\mathbf{x} - \mathbf{x}_0)\right]^{\mathrm{LT}} \tag{3.17}$$

where "LT" denotes the transposition between the two left indices.

We note that even though the condensation method accelerates the calculation of $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$ for each RVE, parallel computations based on it in a multiscale analysis are memory demanding and still quite expensive. Hence, to avoid the online condensation procedure, we utilize a GP to learn the relation between microstructural morphology and effective elastic tangents for different RVEs which are precomputed by the condensation method in an offline stage.

3.4.3 Deflated clustering analysis (DCA)

In a multiscale simulation, the elasto-plastic response of the RVEs associated with the macro-IPs can be obtained via the stabilized micro-damage algorithm (see Section 3.4.1). These computations are very expensive and so we use the DCA method [69] to dramatically accelerate them. Compared to other clustering-based ROMs [68, 70, 73] which primarily speed up micro analyses, our method can accelerate both macro- and micro-simulations. Its high efficiency comes from the fact that (1) the degrees of freedom are significantly reduced from a large number of finite elements to a few clusters by employing material clustering techniques and (2) the algebraic system on the reduced system has much fewer close-to-zero eigenvalues (and hence better convergence behavior) compared to the classic finite element system.

DCA uses clustering to agglomerate neighboring finite elements to a set of interactive irregularly shaped clusters. Clustering is an unsupervised machine learning technique to interpret and group similar data. Among many mature clustering algorithms [74], we adopt k-means clustering [75] in this work due to its simplicity. We start the k-means clustering by feeding the coordinates of element centers into a feature space where cluster seeds are randomly scattered and serve as initial cluster means. Then, we assign each element to the cluster with the closest mean. Meanwhile, cluster shapes are iteratively updated to minimize the within-cluster variance, see an illustration in Figure 3.2. Mathematically, the clustering can be stated as the following minimization problem:

$$\mathbf{C} = \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{I=1}^{k} \sum_{n \in C^I} \|\varphi_n - \bar{\varphi}_I\|^2 \tag{3.18}$$

where $\mathbf{C}$ represents the k-clusters with $\mathbf{C} = \{C^1, C^2, \ldots, C^k\}$. φ_n and $\bar{\varphi}_I$ are the coordinates of the nth element center and the mean of the Ith cluster, respectively. Upon clustering, we construct a reduced mesh by connecting cluster centroids via Delaunay triangularization where topological relations are preserved by checking the connectivity between clusters. We

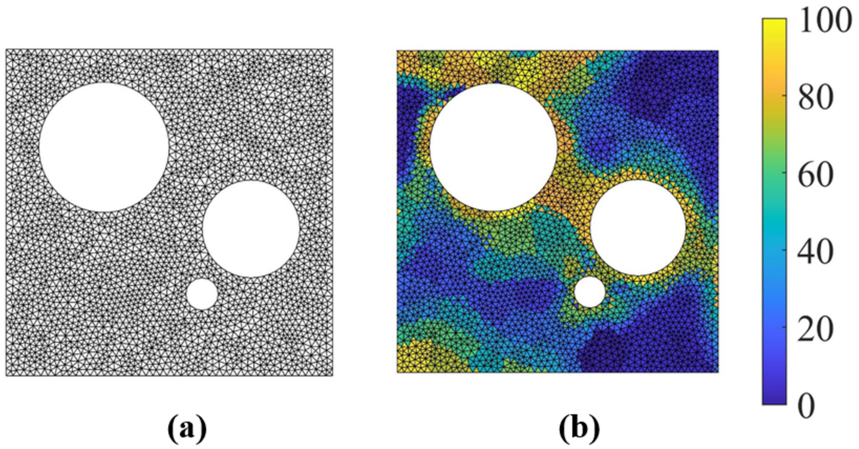

Figure 3.2 Illustration of material points clustering: (a) a generic 2D RVE is discretized with 5000 triangle finite elements and (b) the elements are grouped into 100 clusters via k-means clustering where the elements in the same cluster are indicated by the same color.

assume the motions of cluster centroids are directly related to the grouped nodes. Specifically, the displacement of the cluster centroid $\mathbf{u}(\mathbf{x})$ is computed by interpolating the nodal displacements via the polynomial augmented radial point interpolation method [76] as

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{n} R_i(\mathbf{x}) a_i + \sum_{j=1}^{m} Z_j(\mathbf{x}) b_j \tag{3.19}$$

where a_i is the coefficient of the radial basis function R_i at the ith FE node and b_j is the coefficient of the polynomial basis Z_j. n and m are the number of cluster nodes and the number of polynomial basis functions, respectively. The coefficients a_i and b_j are determined by enforcing Equation (3.19) for all nodal displacements in the cluster where polynomial basis and radial coefficients are assumed to satisfy Equation (3.20) to ensure solution uniqueness [76].

$$\sum_{i=1}^{n} Z_j(\mathbf{x}) a_i = 0, j = 1, 2, \ldots, m \tag{3.20}$$

We then augment the displacements of cluster centroids with rotational degrees of freedom to represent rigid-body motions (three translations and three rotations in 3D) in a deflation space [77, 78, 79] where a reduced stiffness matrix is constructed with six degrees of freedom on each node. Performing nonlinear analyses on the reduced mesh and projecting the results

back to the finite element nodes at the end of computations reads

$$\mathbf{u}_i^j = \mathbf{W}_i^j \boldsymbol{\lambda}_i \tag{3.21}$$

where $\mathbf{u}_i^j$ is the displacement vector at the ith node in the jth cluster, $\boldsymbol{\lambda}_i$ is the rigid-bodymotion of the centroid of the jth cluster, and $\mathbf{W}_i^j$ is the deflation matrix for the ith node group in the jth cluster

$$\boldsymbol{\lambda}_j = [u_{jx}, u_{jy}, u_{jz}, \theta_{jx}, \theta_{jy}, \theta_{jz}]^{\mathrm{T}} \tag{3.22}$$

$$\mathbf{W}_i^j = \begin{bmatrix} 1 & 0 & 0 & 0 & z_i^j & -y_i^j \\ 0 & 1 & 0 & -z_i^j & 0 & x_i^j \\ 0 & 0 & 1 & y_i^j & -x_i^j & 0 \end{bmatrix} \tag{3.23}$$

where u_{jx} and θ_{jx} are the displacement and rotation of the jth cluster along the x-axis, and (x_i^j, y_i^j, z_i^j) are the relative 3D coordinates of the ith node with respect to the centroid of the jth cluster.

We note that material points are assumed to share the same stress and strain values in each cluster. Hence, the local plastic strain fields are reproduced in a diffusive manner with lower strain concentrations which, in turn, delay the onset of localized fracture. This diffusive behavior motivates the damage parameter calibration using LMGP in the next subsection.

3.4.4 Calibration via LMGP

The detailed steps of the proposed framework are included in Algorithm 1. Our LMGP-based data-driven calibration has two major steps which are detailed below and demonstrated in Section 3.5.4.

In the first step, we build the training dataset where the responses (UTS and toughness)

Algorithm 1 Framework of the data-driven calibration for ROM damage parameters via LMGP

procedure ▷ Calibrate the damage parameters of ROMs with different morphologies and fidelity levels
    ▷ DoE variables include pore descriptors, damage model parameters, and simulation fidelity levels
    ▷ Fidelity level = 1, 2, and 3: use ROMs as simulators with different numbers of clusters k.
    ▷ Fidelity level = 4: use DNS as the damage simulator
    ▷ Step-1:
    Set up upper and lower bounds of DoE variables and load DoE
    for $i \leftarrow 1$ to N **do** ▷ Loop over a total of N DoE samples
        Read pore descriptors at the DoE point-i
        ▷ MCR: microstructure characterization and reconstruction
        Reconstruct RVE's geometry via MCR based on pore descriptor values
        Load mesh module to generate FE mesh on the reconstructed RVE geometry
        ▷ CM: condensation method
        Load CM (Section 3.4.2) to compute the effective elastic modulus $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$
        ▷ Assume the effective RVE properties are isotropic
        Compute Lame constants from $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$
        Save the effective Lame constants of the RVE-i
        Read the damage parameters and fidelity level at the DoE sample point-i
        ▷ Damage responses include ultimate tensile strength (UTS) and toughness
        Perform damage analyses (Section 3.4.1)
        if fidelity level $\leftarrow \{1, 2, 3\}$ **then**
            ▷ Use ROM for damage analyses
            Read k and load ROM (k) to compute effective damage responses (Section 3.4.3)
        else if fidelity level $\leftarrow 4$ **then**
            ▷ Use DNS for damage analyses
            Load DNS to compute effective damage responses
        end if
        Save the effective damage responses for each DoE sample point-i
    end for
    ▷ Step-2:
    Read effective damage responses
    Encode damage responses to let LMGP surrogate two damage responses (UTS and toughness)
    Load LMGP (Section 2.2.1) ▷ Consider model fidelity levels as categorical variables
    ▷ Calibrate ROM damage parameters ($\bar{E}^{\mathrm{cr}}$ and α) for different RVE and fidelity level
    for $i \leftarrow 1$ to N **do** ▷ Loop over N RVE samples
        for $j \leftarrow \{1, 2, 3\}$ **do** ▷ Loop over three different ROM fidelity levels
            ▷ Use damage parameters as optimization variables

Minimize the difference of the damage responses between DNS_i and ROM_{ij}
Save the optimal ROM damage parameters toa database
end for
end for
return the database of the calibrated damage parameters of ROMs
end procedure

characterize RVEs' effective softening behavior while the inputs are pore morphology descriptors, damage parameters, and simulation fidelity level. While the latter input is qualitative/categorical and is chosen based on the simulator cost, the other inputs are all quantitative and selected via DoE. For training sample i, we generate the RVE corresponding to the ith set of descriptors via descriptor-based reconstruction techniques. We then deform this RVE via the simulator with the chosen fidelity level which uses the damage parameters of training sample i. Once the training dataset is built, we train the LMGP that simultaneously surrogates all the data sources.

In the second step, we solve an optimization problem to estimate the damage parameters that must be used for an ROM such that it predicts the same UTS and toughness as DNS which uses known material properties (i.e., a fixed set of values for $\bar{E}^{\text{cr}}$ and α) for any RVE. The estimated $\bar{E}^{\text{cr}}$ and α for an ROM depend on the microstructural descriptors of the RVE (numerical inputs) and the ROM's fidelity level (a categorical input). Hence, the objective function of the (inverse) optimization problem measures the difference between predictions of ROM and DNS conditioned on these mixed inputs. Once $\bar{E}^{\text{cr}}$ and α (i.e., the modified material properties) are estimated for each RVE for all ROMs, we conduct multiscale damage analyses where microscale simulations are carried out via ROMs.

3.5 Results

We apply the proposed data-driven framework to calibrate the ROMs in a multi-fidelity and multiscale model that simulates the damage behavior of a metallic component with spatially varying microstructures. In Section 3.5.1, we train a GP to emulate the condensation method to accelerate the online calculation of $\mathbb{C}_{\mathrm{M}}^{\mathrm{el}}$ for each microstructure. In Section 3.5.2, we construct a multi-fidelity model via LMGPs which are then used in Section 3.5.3 to calibrate the damage parameters of ROMs. In Section 3.5.4, we use the calibrated ROMs to investigate the influence of porosity on the structural damage responses of a multiscale model.

The material studied in this work is the cast aluminum alloy A356 whose elastic properties are:

$$Y = 540 \times 10^4 \text{ MPa}, \quad \nu = 0.33 \tag{3.24}$$

where Y and ν are Young's modulus and Poisson's ratio, respectively. The alloy's plasticity is modeled by following the J2 plasticity theory with the piecewise linear hardening curve in Figure 3.3. We assume that plasticity satisfies an associative plastic flow rule with the yield condition as

$$\bar{S} \leq S_Y(\bar{E}^{\mathrm{pl}}) \tag{3.25}$$

where $\bar{S}$, $\bar{E}^{\mathrm{pl}}$ and S_Y are Mises equivalent stress, equivalent plastic strain, and yield stress, respectively.

The softening behavior of A356 is modeled by the progressive damage model in Equation (3.10) with two damage parameters that are applied for all ROMs and DNS: critical plastic strain ($\bar{E}^{\mathrm{cr}}$) and damage evolutionary rate parameter (α). The two damage parameters are selected for calibration in this work because they both significantly affect damage responses; however, more parameters can be calibrated using our proposed framework.

$\bar{E}^{\mathrm{cr}}$ determines the onset of softening that influences the largest stress that a material can

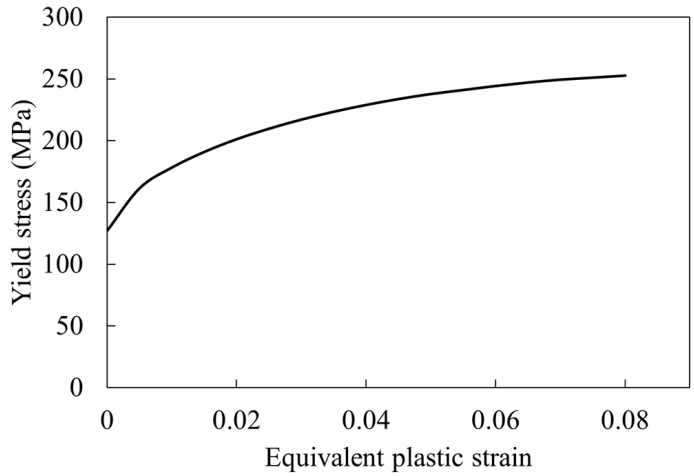

Figure 3.3 Hardening behavior: piecewise linear hardening.

withstand and α controls the amount of released fracture energy which determines the degradation rate of material properties amid damage evolution. The values of damage parameters used in DNS are given in Equation (3.26), while their values for ROMs need to be calibrated based on microstructural morphology and fidelity levels

$$\bar{E}^{\mathrm{cr}} = 0.03 \qquad \alpha = 100 \tag{3.26}$$

Our method is implemented in MATLAB [80] and we obtain the RVE responses on a high-performance cluster paralleled by 40 cores (AMD EPYC processor running at 4.1 GHz) with 120GB RAM.

3.5.1 Gaussian Process Modeling for Microstructure Effective Tangents

As described in Section 3.4.1 and 3.4.2, the effective elastic tangent matrix relates the effective reference stresses with elastic strains. This matrix plays a fundamental role in continuum damage analysis since it enables simulating the progressive fracture evolutions at any IPs in a multiscale model, see Figure 3.1(a). However, computing the effective tangents often

involves intensive computational efforts even when condensation methods are applied, see line 13 of Algorithm 1. Hence, we improve efficiency by developing a GP surrogate that correlates porosity morphology with microstructural effective tangent matrix.

We approximate the complex pores via overlapping ellipsoids whose geometry and spatial distribution are characterized by the following four descriptors: porosity volume fraction V_f, number of pores N_p, aspect ratio between ellipsoidal axes A_r, and the mean nearest distance between centroids $\bar{r}_d$. In addition, as we assume to work with isotropic microstructural responses, the components of the tangent matrix are reduced to two effective Lame constants (μ and λ). In this manner, our GP aims to build a predictive model between $[V_f, N_p, A_r, \bar{r}_d]$ and $[\mu, \lambda]$.

To construct the GP, we first generate a training dataset with 160 RVEs. The inputs in this data set are generated via a DoE where each sample specifies the values of $[V_f, N_p, A_r, \bar{r}_d]$ for each RVE. We let the pore parameters satisfy the ranges in Equation (3.27) where L represents RVE's side length.Then, we use a microstructure reconstruction algorithm [81] to rebuild RVEs corresponding to DoE points. We demonstrate 12 reconstructed microstructures from the DoE datasets in Figure 3.4 where their pore descriptors $[V_f, N_p, A_r, \bar{r}_d]$ are enumerated in Table 3.1. In the reconstructed RVEs, the pore sizes are much smaller than the microstructures which reduce the property variations across different microstructure realizations that have the same four descriptors.

Once the dataset of RVEs is built, we use the condensation method to calculate the effective Lame constants for each RVE. Finally, we train a GP to emulate the relation between $[V_f, N_p, A_r, \bar{r}_d]$ and $[\mu, \lambda]$

$$\begin{cases} 1\% \leq V_f \leq 20\% \\ 10 \leq N_p \leq 100 \\ 1 \leq A_r \leq 5 \\ 0.1L \leq \bar{r}_d \leq 0.5L \end{cases} \tag{3.27}$$

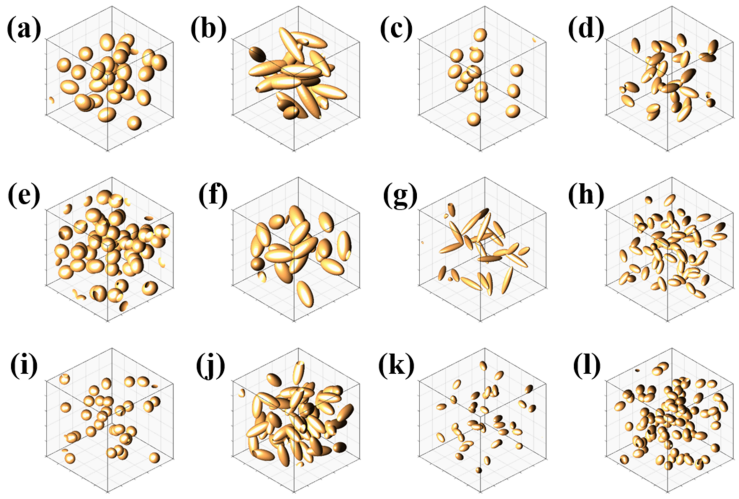

Figure 3.4 12 examples of reconstructed microstructures in (**a**) – (**l**): the values of microscale porosity descriptors and effective Lame constants are listed in the Table 3.1.

RVE	V_f	N_p	A_r	$\bar{r}_d$	$\mu(\times 10^{10})$	$\lambda(\times 10^{10})$
(a)	6.56%	26	1.31	23.3	1.94	3.51
(b)	9.21%	20	3.33	19.7	1.82	3.05
(c)	2.06%	13	1.14	28.1	2.08	3.96
(d)	3.29%	29	2.37	20.5	2.03	3.78
(e)	9.97%	48	1.16	20.4	1.85	3.23
(f)	7.80%	20	2.15	25.9	1.89	3.31
(g)	1.92%	22	4.95	22.4	2.08	3.92
(h)	3.12%	60	2.11	16.9	2.04	3.81
(i)	2.61%	31	1.09	21.6	2.07	3.91
(j)	9.70%	51	2.47	18.2	1.82	3.09
(k)	1.15%	36	1.84	21.1	2.11	4.03
(l)	4.48%	77	1.43	14.5	2.01	4.02

Table 3.1 Pore descriptors and effective Lame constants. Note: the numbers correspond to the reconstructed microstructures in Figure 3.4.

To test the GP's accuracy, we split the data set by using 80% for training and 20% for validation. Comparisons of the predictions with the test samples are shown in Figure 3.5.

To assess the convergence and whether sufficient training data are used, we split the dataset into 100 samples for training and 60 samples for testing. We sequentially increase the size of the training data from 10 to 100 and evaluate the accuracy of the corresponding GPs on 60 test samples (all GPs are evaluated on the same set of test samples). The prediction errors

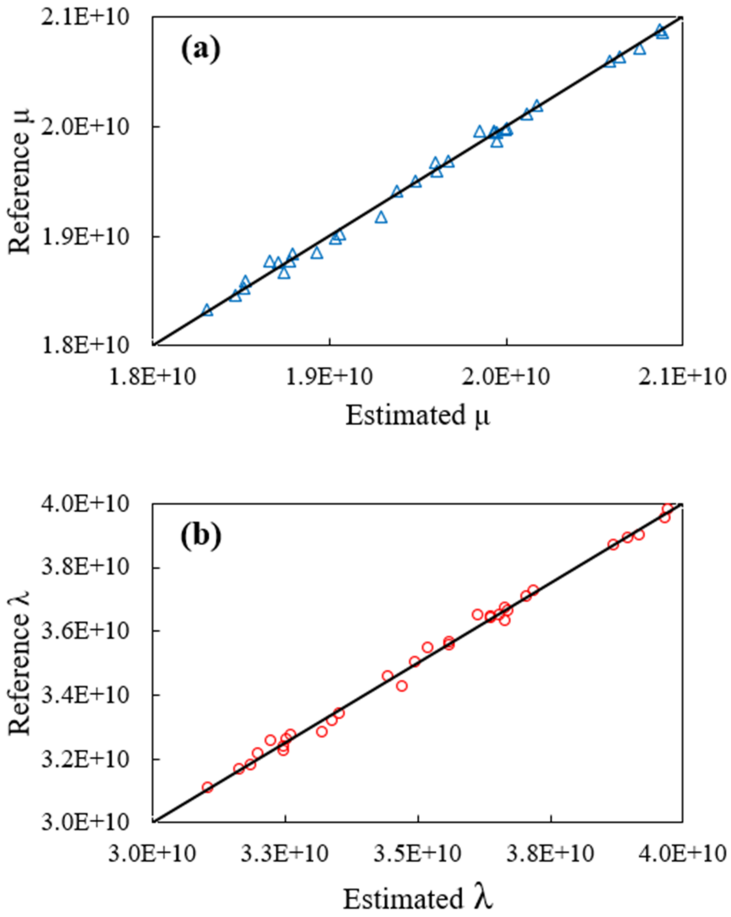

Figure 3.5 Emulation accuracy: comparison of the actual values of the two microstructural effective Lame constants against the GP predictions on unseen test samples in (a) and (b).

are computed by Equation (3.28)

$$E_{\mathbf{y}} = \frac{1}{N_v} \sum_{i=1}^{N_v} \frac{\|\hat{\mathbf{y}}_i - \overline{\mathbf{y}}_i\|}{\|\overline{\mathbf{y}}_i\|} \tag{3.28}$$

where N_v is the number of validation samples, $E_{\mathbf{y}}$ is the relative prediction error of responses $\mathbf{y} = [\boldsymbol{\mu}, \boldsymbol{\lambda}]$, $\hat{\mathbf{y}}_i$ and $\overline{\mathbf{y}}_i$ are the predicted effective Lame constants of the ith RVE. The convergence curve is shown in Figure 3.6 where it is observed that with the increase of training samples, prediction errors monotonically decrease. With 100 samples the prediction error drops to lower than 0.4%, indicating highly accurate predictions. Therefore, we use the GP emulated effective modulus to replace the condensation method amid online computations to accelerate damage analyses for all microstructures.

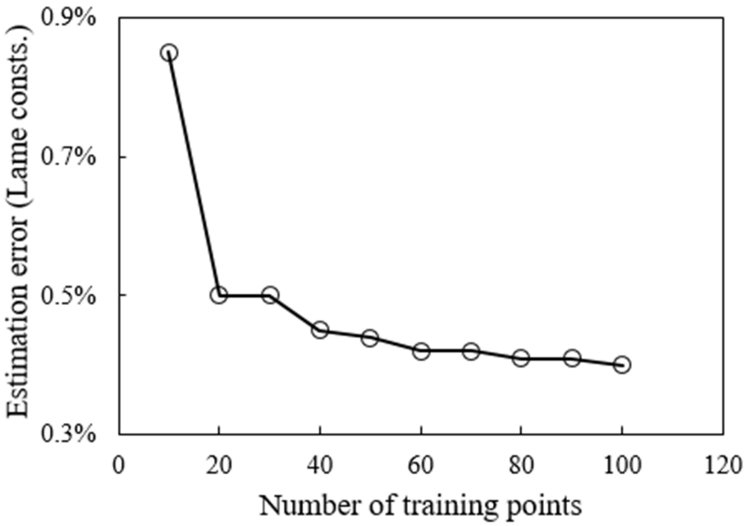

Figure 3.6 Error convergence: GP estimation errors of the predicted Lame constants with respect to the number of training points.

3.5.2 LMGP Modeling of Damage Parameters

In this subsection, we train an LMGP that is used in Section 3.5.3 for ROM calibration, see Figure 3.1(c) where the LMGP is used in the inverse optimization. We first demonstrate the importance of calibration and then provide the details on the training and validation of the LMGP. To demonstrate the importance of parameter calibrations of ROMs, let us consider the microstructure in Figure 3.7(a) with pore descriptors $[V_f, N_p, A_r, \bar{r}_d] = [15.9\%, 25, 1.4, 24.3]$ and damage parameters in Equation (3.26). We subject this RVE to the deformation gradient in Equation (3.29) and compute its responses via the DNS using $68,675$ finite elements as shown in Figure 3.7(b) where significant strain concentrations are observed in the vicinity of pores. We then model this RVE via an ROM with 3200 clusters using the same damage parameters as the DNS. The plastic strain distributions are shown in Figure 3.7(c) which demonstrates the diffusive nature of local clustering

$$\mathbf{F}^{\mathrm{M}} = \begin{bmatrix} 1.1 & 0 & 0 \\ 0 & 0.95 & 0 \\ 0 & 0 & 0.95 \end{bmatrix} \tag{3.29}$$

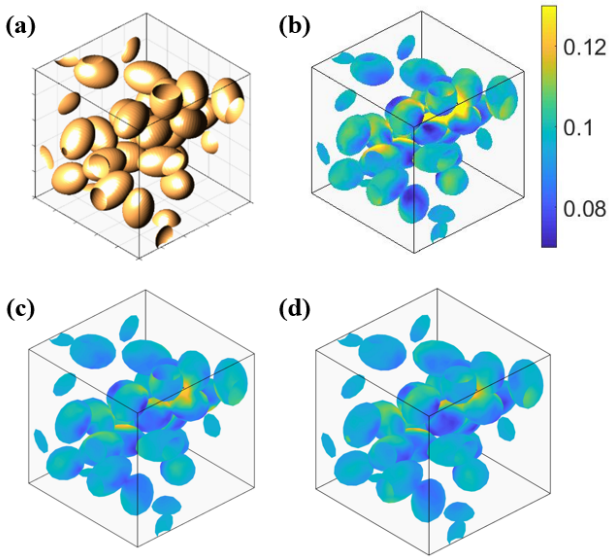

Figure 3.7 Equivalent plastic strain fields: (a) the porosity morphology of a microstructure with 25 pores, (b) plastic strains are simulated via DNS, (c) plastic strains are approximated by ROM ($k = 3200$) without calibration, and (d) plastic strains are approximated by ROM ($k = 3200$) with calibration.

Comparison of strain distributions in Figure 3.7 demonstrates that the label of the data source (i.e., DNS or ROM), which we consider as a categorical variable in LMGP, must encode the diffusive nature of the local solutions. Additionally, compared to the DNS, the clusterwise solutions of ROMs have lower magnitudes of plastic strains which result in delayed fracture initiation, higher UTS, and larger material toughness, see Figure 3.8(a) and Table 3.2.

The accuracy of ROMs can be improved by calibrating their damage parameters ($\bar{E}^{\text{cr}}$ and α). We illustrate calibration effects on the local strain concentrations and effective behav-

	Precalibration		After calibration	
Simulation fidelity	UTS	Toughness	UTS	Toughness
DNS	1.03×10^8	3.71×10^6	-	-
$k = 800$	1.15×10^8	4.10×10^6	1.07×10^8	3.85×10^6
$k = 1600$	1.09×10^8	3.93×10^6	1.043×10^8	3.783×10^6
$k = 3200$	1.06×10^8	3.85×10^6	1.046×10^8	3.776×10^6

Table 3.2 Damage responses. Note: values of the UTS and toughness of DNS and ROMs for the microstructure in Figure 3.7.

iors in Figure 3.7(d) and 3.8(b), respectively. It is evident that compared to the ROMs with the damage parameters of DNS, the calibrated ROMs provide more accurate and effective stress–strain responses. Specifically, the calibration algorithm calibrates (i.e., reduces) the ROMs' critical plastic strain to induce early softening so that their UTS values become closer to that observed in DNS. Meanwhile, the calibration also decreases the ROMs' damage evolutionary rate parameters to compensate for the toughness reduction due to early softening.

We further compare the values of the material toughness and UTS between DNS and ROMs in Table 3.2 where we find that the accuracies of both damage responses are improved after calibrations. The improvement is demonstrated by the enumerated errors in Table 3.3 where we observe that the calibrations significantly reduce the ROMs' model errors ($\mathbf{r}$) for all ROM fidelity levels ($k = 800, 1600, 3200$). Additionally, the magnitudes of the normalized errors $\mathbf{r} = (r_{\text{toughness}}, r_{\text{UTS2}})$ continuously drop with the increase of clusters, which validates our observation in Figure 3.8 that the ROM with more clusters provides closer solutions to the DNS in both pre and post-calibration scenarios.

We provide the values of the calibrated damage parameters in Table 3.4. We note that with the decrease of simulation fidelity levels from DNS to the ROMs ($k = 3200, 1600, 800$), both values of $\bar{E}^{\text{cr}}$ and α decrease. This trend implicitly validates our previous observation in Figure 3.7(b) and 3.7(c) that fewer clusters result in more diffusive cluster-wise plastic strains with delayed damage initiations in Figure 3.8(a). This is because, to counteract the

	Errors ($\boldsymbol{r}$) w/o LMGP calibration (%)			Errors ($\boldsymbol{r}$) w. LMGP calibration (%)		
ROM clusters (k)	UTS	Toughness	$\|\boldsymbol{r}\|$	UTS	Toughness	$\|\boldsymbol{r}\|$
800	11.5	10.6	15.61	3.74	3.64	5.22
1600	5.5	5.8	7.99	1.28	1.96	2.34
3200	3.2	3.7	4.86	1.55	1.73	2.33

Table 3.3 ROM prediction error. Note: errors of ROMs on UTS and toughness for the microstructure in Figure 3.7.

Simulation fidelity	$\bar{E}^{\text{cr}}$	α
ROM ($k = 800$)	0.021	36.31
ROM ($k = 1600$)	0.024	47.65
ROM ($k = 3200$)	0.027	72.27
DNS	0.030	100

Table 3.4 Calibrated damage parameters. Note: values of calibrated ROM damage parameters for the microstructure in Figure 3.7.

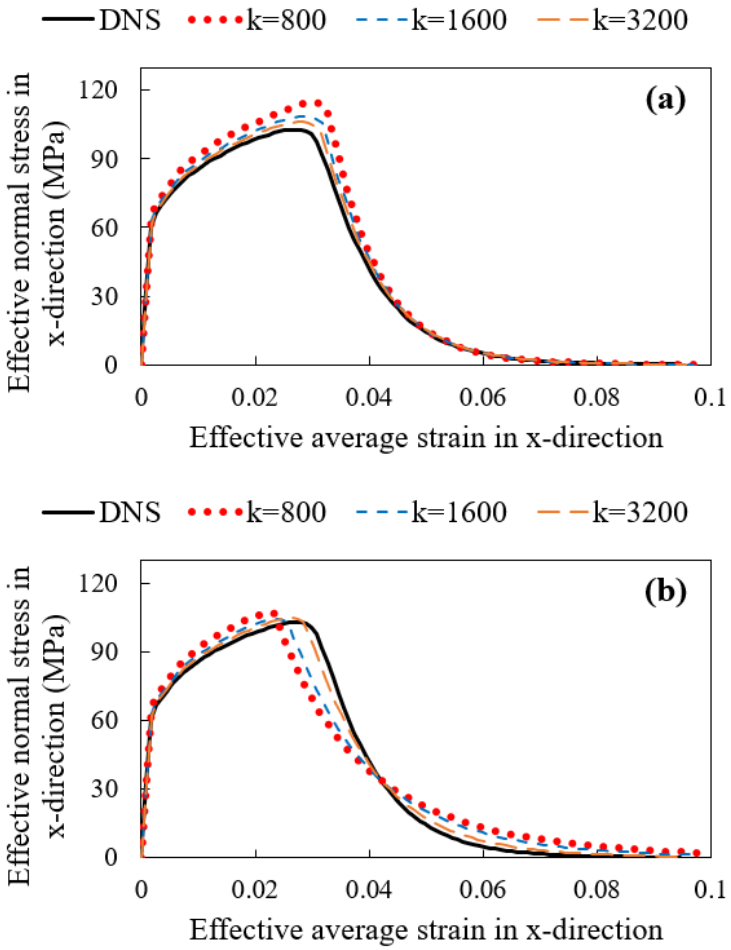

Figure 3.8 Importance of calibration: (a) the effective stress–strain curves without damage parameters calibration and (b) the effective response with calibration.

artificial delay of softening, the calibration algorithm needs to lower $\bar{E}^{\text{cr}}$ so that softening occurs at smaller deformation conditions. Meanwhile, the calibration decreases α to lower the material's degradation rates during damage evolutions which helps to approximate ROMs' toughness to that of DNS.

Once the accuracy of ROMs is improved via calibration, they can substitute DNS. We compare the computational time of DNS against the ROMs in Figure 3.9 by CPU time. We observe that while DNS takes 29.8 h to finish the damage simulation, it only takes about 68.8, 27.9, and 15.6 min for the ROMs with 3200, 1600, and 800 clusters, respectively. The ROMs' computational costs can be further reduced to online time since their offline stages (clustering and preprocessing) are performed only once and are unnecessary in future calculations. Thus, by comparing the online time with DNS, the acceleration factors of the ROMs with 3200, 1600, and 800 clusters are 44.1, 99.9, and 242.9, respectively. After demonstrating the necessity of ROM calibration, we now describe the proposed LMGP-based calibration approach. Compared to manual calibrations, the proposed data-driven approach is highly efficient in automatically allocating the optimal values of the damage parameters for the ROMs based on their fidelity levels as well as the microstructure.

To use LMGP for calibration, we generate a data set consisting of six inputs $\mathbf{x} = [x_1 \ldots, x_6]^T$ and two outputs $\mathbf{y}$, as shown in Table 3.5. The first four inputs represent the pore descriptors (i.e., $[V_f, N_p, A_r, \bar{r}_d]$) and the last two inputs represent the two damage parameters (i.e., evolutionary rate parameter α and critical effective plastic strain $\bar{E}^{\text{cr}}$). We generate DoE sample points via Sobol sequence by satisfying the ranges of descriptor values and damage parameters in Equation (3.27) and 3.30, respectively. Two LMGP outputs are the two

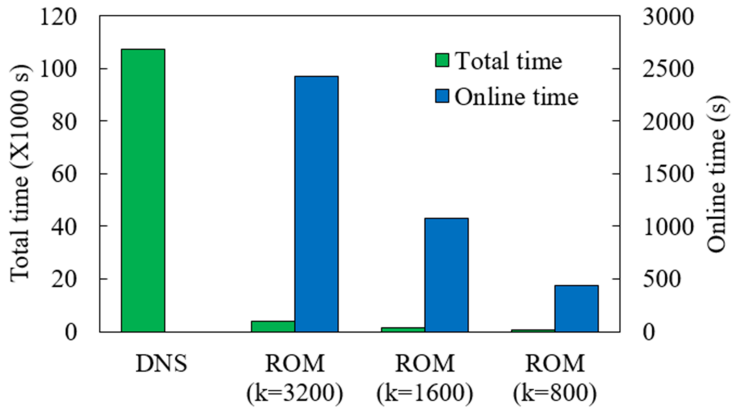

Figure 3.9 Time reduction: computational time comparison between DNS with ROMs.

damage responses (UTS and material toughness)

$$\begin{cases} 1\% \leq \bar{E}^{\text{cr}} \leq 3\% \\ 10 \leq \alpha \leq 100 \end{cases} \tag{3.30}$$

We append each sample point with a categorical variable to encode the data source which is denoted by $t_1 = \{1, 2, 3, 4\}$ where label 4 corresponds to DNS while labels 3, 2, and 1 correspond to the ROM with $k = 3200$, $k = 1600$, and $k = 800$ respectively. To enable

$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	$\boldsymbol{x}_3$	$\boldsymbol{x}_4$	$\boldsymbol{x}_5$	$\boldsymbol{x}_6$	$\boldsymbol{t}_1$	$\boldsymbol{t}_2$	$\boldsymbol{y}$
0.021	13	1.14	28.1	54.7	0.015	4	1	1.12×10^8
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.066	26	1.31	23.3	71.2	0.017	4	1	1.15×10^8
0.098	87	1.89	12.4	75.6	0.020	3	1	1.13×10^8
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.045	77	1.43	14.5	80.7	0.023	3	1	1.26×10^8
0.030	70	3.93	12.6	73.4	0.066	2	1	1.21×10^8
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.026	31	1.10	21.6	98.3	0.029	2	1	1.33×10^8
0.078	34	2.77	17.4	21.3	0.012	1	1	1.08×10^8
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.016	88	3.13	14.4	61.7	0.027	1	1	1.36×10^8
0.021	13	1.14	28.1	54.7	0.015	4	2	3.14×10^6
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.067	26	1.31	23.3	71.2	0.017	4	2	3.00×10^6
0.098	87	1.89	12.4	75.6	0.020	3	2	3.26×10^6
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.045	77	1.43	14.5	80.7	0.023	3	2	3.93×10^6
0.030	70	3.93	12.6	73.4	0.066	2	2	3.07×10^6
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.026	31	1.10	21.6	98.3	0.029	2	2	4.72×10^6
0.078	34	2.77	17.4	21.3	0.012	1	2	3.17×10^6
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0.016	88	3.13	14.4	61.7	0.027	1	2	5.05×10^6

Table 3.5 Training data set of LMGP: four microstructure descriptors ($\boldsymbol{x}_1 \sim \boldsymbol{x}_4$), two damage parameters ($\boldsymbol{x}_5 \sim \boldsymbol{x}_6$), and two categorical inputs ($\boldsymbol{t}_1 \sim \boldsymbol{t}_2$) which encode data source and response type.

LMGP to simultaneously surrogate two damage responses, we also appended the samples with a second categorical variable encoding the type of responses by $t_2 = \{1, 2\}$ where label 1 corresponds to UTS and label 2 indicates material toughness. Part of the resulting single-response training dataset is shown in Table 3.5. Our entire data set contains 600 samples that are created from four different fidelity sources: 70, 110, 170, and 250 samples are from DNS and the ROMs with 3200, 1600, and 800 clusters respectively.

To investigate the effects of sample sizes on prediction accuracy, we fit our LMGP to 100, 200, 300, and 400 samples and test its performance on 200 testing samples across 50 random repetitions. We note that our data set is unbalanced because we have fewer samples from high-fidelity source that requires high computational costs and much more data points from low-fidelity models. In particular, 10% of the training samples are obtained from DNS, 20% from ROM with 3200 clusters, 30% from ROM with 1600 clusters, and 40% from ROM with 800 clusters. These ratios are the same across all sub-data sets that are created using the entire data set.

We scale LMGP outputs to $[0, 1]$ and compute the mean absolute errors (MAE) of predictions as shown in Figure 3.10. We observe that with the increase of training samples, both MAE and its variance decrease. Therefore, we choose 400 samples as our training data set which contains 40 samples of DNS, 80, 120, and 160 samples from the ROMs with 3200, 1600, and 800 clusters, respectively. Based on users' computational budgets, various combinations of different fidelity sources can be explored. Minimizing the costs of training data sets for multi-fidelity models is, however, out of the scope of this work.

From Figure 3.10, we also notice that the scale of the vertical axis in Figure 3.10(b) is smaller than that of Figure 3.10(a), suggesting that our LMGP provides better predictions for toughness than UTS (we elaborate on the underlying reasons below).

Once LMGP is trained, we can visualize the learned latent space where each combination

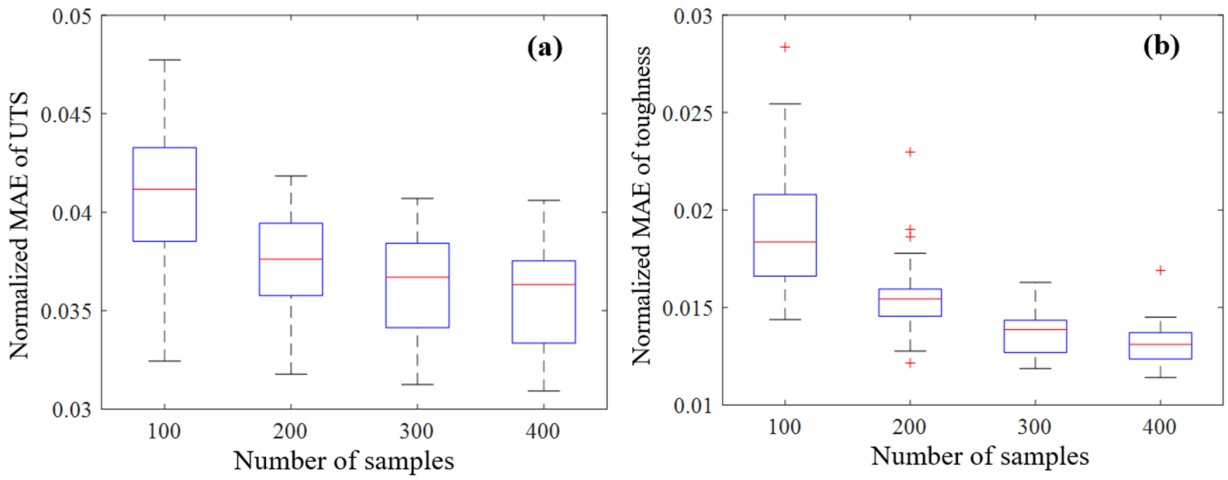

Figure 3.10 LMGP's MAEs: normalized MAE of UTS and toughness with respect to different numbers of training samples.

of the two categorical variables is mapped to a latent position in Figure 3.11. Based on the latent points of the underlying (fidelity level, response) combinations, it is evident that latent axes z_1 and z_2 encode, respectively, the types of damage response and the simulation fidelity levels (note that this encoding is learned automatically by LMGP). We observe that the scale of z_1 is one order of magnitude larger than z_2, suggesting that the latent points are primarily grouped by their damage responses (z_1). For the same damage response, the latent points are further distinguished by their fidelity levels (z_2). Specifically, we find that the positions of $k = 3200$ are far from $k = 800$ but close to DNS, suggesting the damage responses predicted by ROMs with 3200 clusters share more similarity with the DNS than the ROMs with only 800 clusters. This also indicates that low fidelity models (e.g., $k = 800$) exhibit model form error compared to the higher-fidelity sources.

LMGP provides significant insights and interpretations of the characteristics of the studied datasets in Figure 3.11. For example, the vertical distance between DNS and $k = 800$ along the z_2 direction is about 0.06; suggesting a correlation value of ($e^{-0.062} = 0.9964$) between the two data sources, see Equation (2.2). Given this correlation, LMGP can use the knowledge from the low-fidelity data (i.e., $k = 800$) to improve its accuracy in emulating the high-

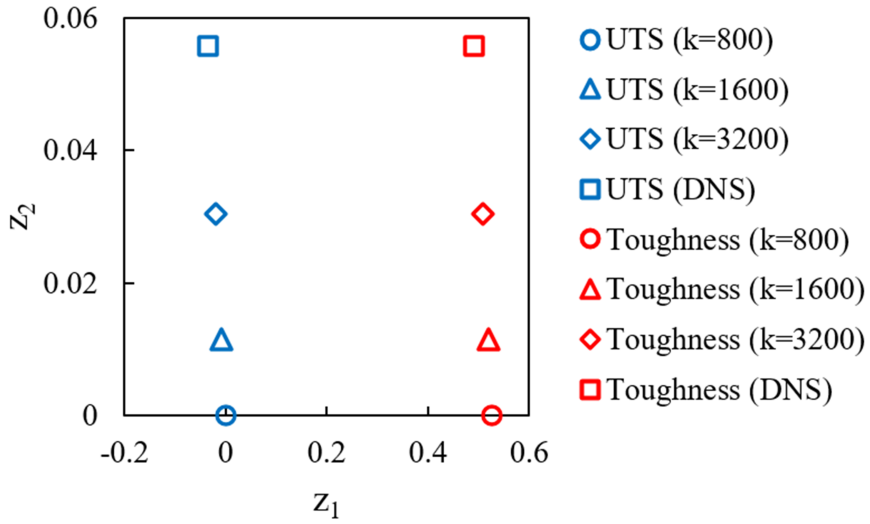

Figure 3.11 Learnt latent space of LMGP: each latent position encodes simulation fidelity level and damage response.

fidelity source (i.e., DNS). We also notice that the horizontal distance along the z_1-axis is about 0.6, resulting in the correlation value of $e^{-0.62} = 0.6977$. It suggests the two responses are positively correlated, consistent with our expectation that the delayed fracture initiation from ROMs' diffusive local solutions not only increases predicted UTS but also enlarges material toughness, see the discussion in Figure 3.8(a).

To assess LMGP's accuracy, we split the 600 sample points into training and validation sets where 400 samples are used for training and the remaining 200 samples are for validation. The validation dataset contains 20, 40, 60, and 80 samples from DNS, the ROM with $k = 3200$, 1600, and 800, respectively. LMGP's prediction accuracy is quantified by the MAE in Table 3.6 where it is observed that the prediction errors are higher for the highest-fidelity source (DNS) as well as the lowest-fidelity ROM with 800 clusters compared to the other two fidelity sources. The reason for the large prediction errors on DNS comes from its data scarcity, and the errors of $k = 800$ are due to its inherent model errors.

We plot LMGP's predictions against validation samples for the two damage responses in Figure 3.12 where the predictions of both responses are found to be quite accurate. Specifically, we notice that the predictions of UTS have higher errors than those of toughness. This

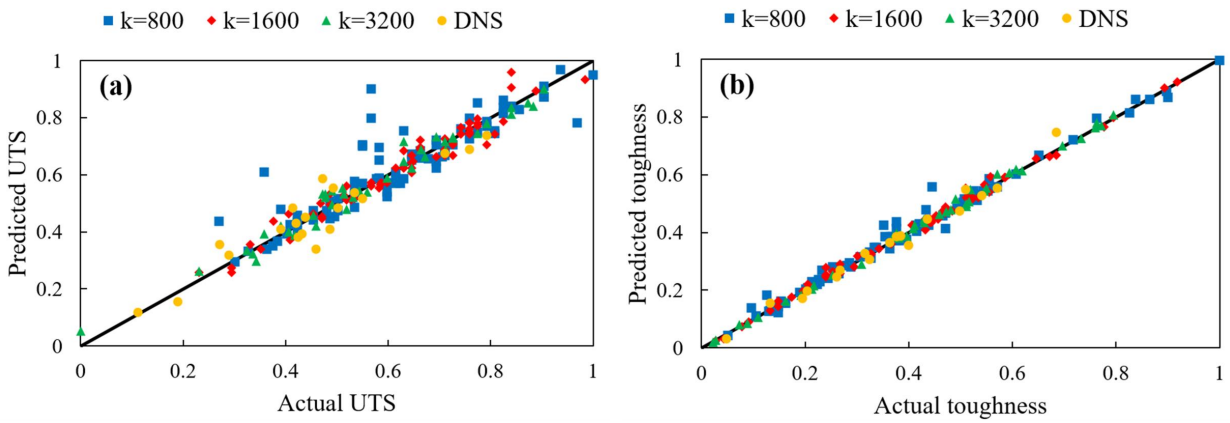

Figure 3.12 Performance on unseen test data: comparison of the true responses against the LMGP's predictions for (a) UTS and (b) toughness.

	MAE	
Simulation fidelity	UTS	Toughness
ROM with $k = 800$	0.0430	0.0152
ROM with $k = 1600$	0.0277	0.0104
ROM with $k = 3200$	0.0274	0.0100
DNS	0.0444	0.0283

Table 3.6 Error analysis: MAE of the LMGP's prediction for the two damage responses and four data sources.

observation is consistent with our discussions in Figure 3.10. One plausible reason is that UTS, as a point measurement of the maximum stress that an RVE can tolerate, is sensitive to some important factors that are not considered in this surrogate, e.g., the directions of crack propagations. In contrast, RVE toughness, which is a global estimation for the amount of released fracture energy amid damage evolution (which is an integral quantity), is characterized by our model quite well.

3.5.3 Calibration of Damage Parameters

To improve solution accuracy, the damage parameters of ROMs need to be calibrated as shown in Figure 3.1(c). We perform the calibration by solving an inverse optimization problem whose objective function is evaluated via LMGP. We estimate the calibration parameters

for the ith microstructure and the jth source level such that the estimated damage responses from ROM match the ones from DNS that uses the true damage parameters: $\alpha_{DNS} = 100$ and $\bar{E}^{\text{cr}}_{\text{DNS}}$. The optimization problem is hence formulated as

$$\left[\hat{\alpha}, \bar{E}^{\text{cr}}\right] = \underset{\alpha, \bar{E}^{cr}}{\arg\min} \left\| \boldsymbol{y}_p\left(\boldsymbol{x}^i_{\text{DNS}}\right) - \boldsymbol{y}_p\left(\boldsymbol{x}^i_j\right) \right\|^2 \tag{3.31}$$

where $\boldsymbol{y}_p$ are the predicted damage responses by LMGP and $\boldsymbol{x}^i_{\text{DNS}} = [V^i_f, N^i_p, A^i_r, \bar{r}^i_d, \alpha_{\text{DNS}}, \bar{E}^{\text{cr}}_{\text{DNS}}, t_1 = 4, \mathbf{t}_2]$ represents the input vector of the ith microstructure for predicting the responses of DNS. $\boldsymbol{x}^i_j = [V^i_f, N^i_p, A^i_r, \bar{r}^i_d, \alpha, \bar{E}^{\text{cr}}, t_1 = j, \mathbf{t}_2]$ is the input vector of the ith microstructure for predicting the damage responses for ROM at the jth fidelity level (note that we pass $\mathbf{t}_2$ as a vector to get both damage responses). We use a gradient-based optimization method to solve Equation (3.31).

In Figure 3.13, we demonstrate the values of the calibrated damage parameters for the 600 microstructures in the database. We note that the calibration is performed based on an inverse optimization which tends to minimize the difference between the damage responses of DNS and ROM where both are surrogated by LMGP, that is, no online microstructure simulation is performed for calibration. Since the optimization relies on an inexpensive surrogate, its computational cost is very small.

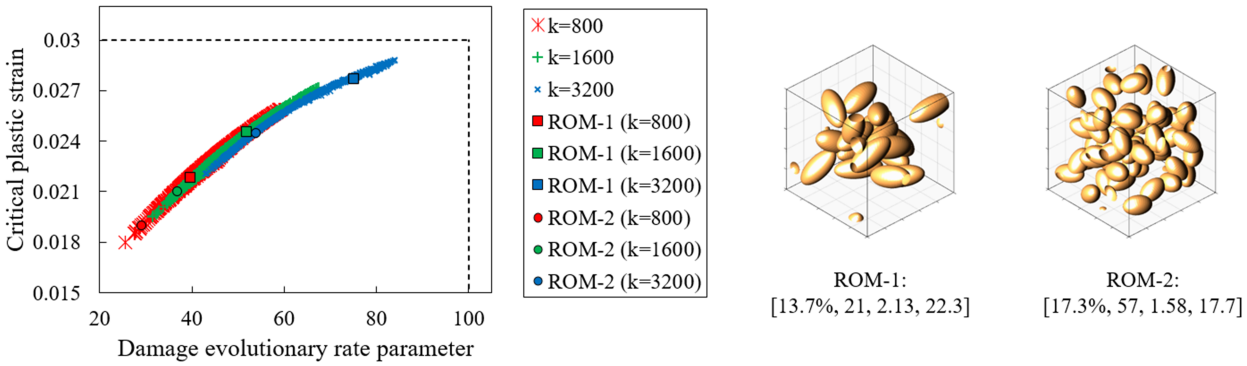

Figure 3.13 Calibrated damage parameters: calibrated damage parameters of 600 samples simulated by ROMs with three fidelity levels where two RVEs with distinct pore morphologies are highlighted.

From Figure 3.13, we observe the same trend across all samples. Specifically, we highlight the calibrated damage values of two distinct RVEs that were not used in training the LMGP. We find that (1) the values of the ROM's calibrated damage parameters are smaller than those of DNS (represented by dashed lines) and (2) the values of calibrated damage parameters are closer to the DNS' values as we increase the number of clusters (k). For instance, the calibrated parameters of both RVEs with 800 clusters are much smaller than their counterparts in DNS or ROMs with 1600 or 3200 clusters. The underlying reason is that as k decreases, the ROM's local plastic strain becomes more diffusive which delays damage initiation and artificially increases UTS and toughness. Therefore, to counteract this diffusive behavior, the calibrated damage parameters tend to reduce the strength of the materials to induce early damage such that the ROMs can faithfully approximate DNS.

3.5.4 Multiscale Damage Analyses

Since manufacturinginduced porosity significantly affects material properties [82, 83, 84], in this section we can apply the reduced multiscale damage model to a 3D L-shape bracket to quantify the impact of microporosity on the bracket's fracture behavior. Our simulations follow Figure 3.1(d) where the calibrated ROMs are used to accelerate the microscale analyses in the multiscale model.

The dimensions of the L-bracket are shown in Figure 3.14. The bracket is fixed on the top surface, and it is subject to a Dirichlet boundary condition on the right surface ($d = 20$ mm). The bracket model is discretized with 2113 linear tetrahedron elements with reduced integrations.

For multiscale analysis, we divide the bracket into two subdomains: a monoscale region and a multiscale region with spatially varying porosity distribution. This choice is motivated by the observation that under large deformations the fracture happens in the multiscale domain

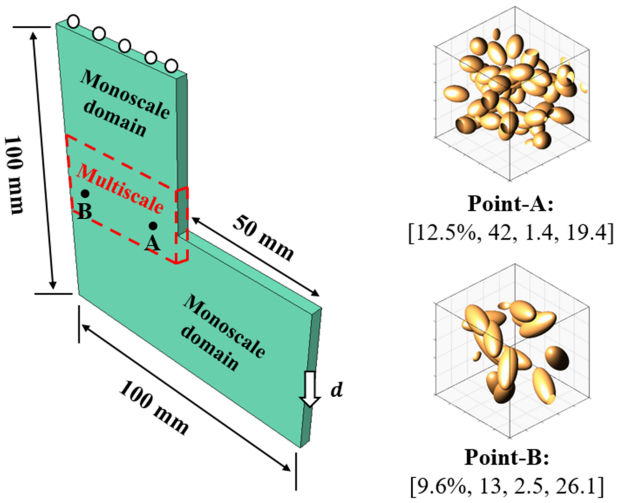

Figure 3.14 Multiscale model: the dimensions and boundary conditions of a 3D L-shape bracket model with a thickness of 5 mm where two RVEs with distinct pore descriptors are associated with two macroscale IPs in the multiscale domain.

where high accuracy and microstructural effects are needed, and hence the other regions of the bracket can be modeled as a single scale.

For each of the 147 IPs in the multiscale region, we randomly assign a microstructure from the database generated in Section 3.5.2. The effective damage behavior in each microstructure is simulated by ROMs with three fidelity options: 800, 1600, or 3200 clusters. For each ROM with a selected cluster number, its optimal damage parameters are readily available from the LMGP-based calibration process described in Section 3.5.3. Specifically, among the 147 macro-IPs, 77 IPs are associated with the RVEs simulated by 800 clusters, 50 and 20 IPs are assigned to the RVEs with 1600 and 3200 clusters, respectively. We note that the RVEs with higher numbers of clusters are assigned to the IPs with anticipated softening that is predicted by a preliminary single-scale simulation without micro-pores, see the two highlighted RVEs with distinct local pore morphologies that are assigned to different IPs in the multiscale region in Figure 3.14.

In our multiscale simulations, we ensure the released fracture energy is consistent between the scales by equating microstructure volumes to macroscopic mesh sizes. Additionally, we

apply a nonlocal damage function with a feature size of 15 mm on the bracket model to prevent pathological mesh dependency and convergence difficulty.

We demonstrate the simulated fracture pattern, local plastic strain distributions, and load-displacement response (with and without multiscale treatment) in Figure 3.15. In Figure 3.15(a), we demonstrate the macro-fractures by elements' effective damage values DM in Equation (3.12) where $D_{\mathrm{M}} = 1$ represents complete material ruptures. We notice that the highlighted two macro-IPs are located in the damage zone, and we plot the distributions of microscopic equivalent plastic strains in Figure 3.15(b) exhibiting significant local strain concentrations. Specifically, we observe that large plastic strains are accumulated in proximity to pore surfaces in the two RVEs which cause the macroscale fractures in Figure 3.15(a). In Figure 3.15(c), we observe that porous microstructures significantly deteriorate the bracket's load-carrying capacity which drops by 10.22% from 70.86 N to 63.62 N, and the bracket breaks at a much lower displacement boundary condition. Therefore, compared to the single scale model that only considers dense materials and neglects pores, the multiscale model provides us with a more realistic prediction by considering fractures across scales.

Our multiscale simulation is paralleled by 40 CPU cores on a high-performance HPC, and it is finished in 15.2 h. Based on the efficiency comparison between the ROM and DNS in Figure 3.9, the estimated computational time for DNS (classic FE2) is more than 2623.5 h (109.3 days), that is, our calibrated ROM speeds up the overall computation compared to DNS with an acceleration factor of 172.6.

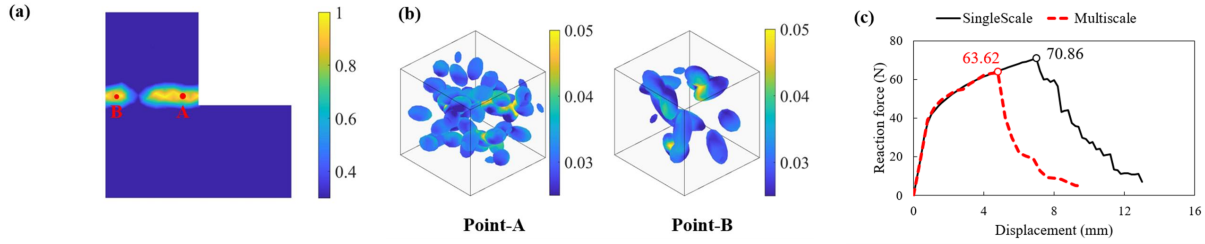

Figure 3.15 Results of multiscale damage analysis: (*a*) the top view of the fracture patterns on the L-bracket model, (*b*) the distributions of equivalent plastic strains in the two highlighted RVEs, and (*c*) the force–displacement responses.

3.6 Conclusions

We propose a multi-fidelity reduced-order model for multiscale damage analysis that considers manufacturing-induced spatially varying porosity. Our model is not only significantly faster than multiscale simulations based on the FE^2 approach but also has lower memory requirements. Our approach relies on a mechanics-based ROM that accelerates the microscale elasto-plastic-damage deformations by clustering the degrees of freedom. Since this clustering artificially increases microstructures' tolerance to damage initiation and evolution, we develop a calibration scheme to estimate the damage parameters that must be used in ROM such that it can faithfully approximate high-fidelity simulations.

We employ LMGPs to build a multi-fidelity emulator which is then used in our calibration scheme. In addition to providing high accuracy and versatility for emulation, we show that the learned latent space of LMGP is interpretable and provides insights into the problem (e.g., determining the relative accuracy of multiple ROMs with respect to DNS). This LMGP-based calibration scheme differs from existing calibration works such as Refs. [85, 86] which focus on calibrating simulations using experimental data. In contrast to these works, we focus on calibrating ROMs against DNS such that these ROMs can be used in multiscale simulations where microstructural details vary over the macro-component.

In this work, we use the calibrated ROMs in a multiscale simulation to study the effect of spatially varying micro-porosity on the macroscopic response of an L-bracket model. Our results indicate that porosity noticeably decreases the strength of the material and hence must be considered in "design for fracture".

In this work, we neglect the inherent uncertainty in material properties, i.e., our simulations (based on either ROMs or DNS) are deterministic. More realistic fracture modeling requires embedding uncertainty sources in our calibration scheme and multiscale simulations. With this treatment, we will obtain probabilistic distributions for the calibrated parameters (con-

ditioned on a selected fidelity level). These microscale distributions are spatially correlated at the macroscale and quantifying their effects on the macroscale quantities relies on sampling technique [87] (e.g., based on Markov chain Monte Carlo). We believe our ROMs provide a unique opportunity for such sampling-based multiscale uncertainty quantification and plan to pursue this direction in our future works.

The proposed data-driven multi-fidelity damage model in this paper opens up some interesting future research directions. For example, the ROMs with calibrated damage parameters can efficiently generate material response databases correlating intricate microstructural morphologies with effective material behaviors under complex loading conditions. Such databases enable deep learning-based surrogates for a direct mapping between material local morphology and their responses for computationally demanding nonlinear analyses. In addition, applying our multi-fidelity model to investigate the effects of material uncertainty on structural behaviors is vital for robust designs as engineered material systems are inherently embedded with manufacturing-induced uncertainties that propagate across scales.

Chapter 4

Final remarks

In this thesis we present two works in the field of data fusion and its application in engineering.

In Chapter 2, we introduce Pro-NDF, a novel NN architecture that converts MF modeling into a nonlinear manifold learning problem. Pro-NDF is based on a multi-block neural network where each block is designed to take on specific tasks for MF modeling problems that arise in typical engineering applications. The main contributions of this work are:

- We present a novel NN architecture for data fusion that can accommodate an arbitrary number of data sets (without requiring prior knowledge of their fidelities) and quantify both epistemic and aleatoric uncertainties.
- We validate each of the key components of Pro-NDF by performing an ablation study on an analytic and a real-world example. In particular, we show that a probabilistic setting not only improves the performance of the emulator but also allows us to develop a novel loss function (based on proper scoring rules) that enhances its robustness.
- We show the usefulness of the fidelity manifold learned by Pro-NDF, which encodes

source-wise similarities/discrepancies. In this manifold, the distance between the encoded sources provides a global measure of their correlation. Thus, it can be used to detect very accurate (or inaccurate) LF sources with respect to the HF source.

- Through a host of multi-source problems, we demonstrate that Pro-NDF outperforms other NN-based data fusion approaches by a large margin. Moreover, Pro-NDF performs on par to LMGP in cases with small data sets and shows better scalability as the size of the training data increases.

In Chapter 3, we present a multi-fidelity reduced-order model for multiscale damage analysis that considers manufacturing-induced spatially varying porosity. The proposed approach relies on a mechanics-based ROM that accelerates the microscale elasto-plastic-damage deformations by clustering the degrees of freedom. Since this clustering artificially increases microstructures' tolerance to damage initiation and evolution, we develop a MF calibration scheme based on LMGPs to estimate the damage parameters that must be used in ROM such that it can faithfully approximate high-fidelity simulations. The main contributions of this work are:

- We open the doors for the fracture-aware design of multiscale materials by proposing a data-driven framework that integrates a mechanistic ROM with a MF calibration scheme based on LMGPs.

- We show that the proposed method is significantly faster than multiscale simulations based on the FE^2 approach.

- We demonstrate the effectiveness of LMGP for MF calibration tasks. We also show that the learned latent space of LMGP is interpretable and enables to determine the relative accuracy of the ROMs with respect to DNS.

- We validate the application of our MF framework in predicting the damage behavior

of a multiscale metallic component with spatially varying porosity. The results show that porosity noticeably decreases the strength of the material and hence must be considered in the design process.

The contributions made in this thesis suggest several promising areas for further investigation, which are discussed in Section 2.5 and 3.6.

Bibliography

[1] Ghanshyam Pilania, James E Gubernatis, and Turab Lookman. Multi-fidelity machine learning models for accurate bandgap predictions of solids. *Computational Materials Science*, 129:156–163, 2017.

[2] Shiguang Deng, Carlos Mora, Diran Apelian, and Ramin Bostanabad. Data-driven calibration of multifidelity multiscale fracture models via latent map gaussian process. *Journal of Mechanical Design*, 145(1):011705, 2023.

[3] Xiaotong Liu, Pierre-Paul De Breuck, Linghui Wang, and Gian-Marco Rignanese. A simple denoising approach to exploit multi-fidelity data for machine learning materials properties. *arXiv preprint arXiv:2204.10430*, 2022.

[4] Shiguang Deng, Diran Apelian, and Ramin Bostanabad. Adaptive spatiotemporal dimension reduction in concurrent multiscale damage analysis. *Computational Mechanics*, 2023.

[5] Zahra Zanjani Foumani, Mehdi Shishehbor, Amin Yousefpour, and Ramin Bostanabad. Multi-fidelity cost-aware bayesian optimization. *Computer Methods in Applied Mechanics and Engineering*, 407:115937, 2023.

[6] Souvik Chakraborty, Tanmoy Chatterjee, Rajib Chowdhury, and Sondipon Adhikari. A surrogate based multi-fidelity approach for robust design optimization. *Applied Mathematical Modelling*, 47:726–744, 2017.

[7] Péter Zénó Korondi, Mariapia Marchi, Lucia Parussini, and Carlo Poloni. Multi-fidelity design optimisation strategy under uncertainty with limited computational budget. *Optimization and Engineering*, 22(2):1039–1064, 2021.

[8] Ghina N Absi and Sankaran Mahadevan. Multi-fidelity approach to dynamics model calibration. *Mechanical Systems and Signal Processing*, 68:189–206, 2016.

[9] Sanaz Zanjani Foumani, Mehdi Shishehbor, Amin Yousefpour, and Ramin Bostanabad. Multi-fidelity cost-aware bayesian optimization. *Available at SSRN 4268166*, 2022.

[10] Siyu Tao, Daniel W Apley, Wei Chen, Andrea Garbo, David J Pate, and Brian J German. Input mapping for model calibration with application to wing aerodynamics. *AIAA journal*, 57(7):2734–2745, 2019.

[11] Slawomir Koziel, Qingsha S Cheng, and John W Bandler. Space mapping. *IEEE Microwave Magazine*, 9(6):105–122, 2008.

[12] John W Bandler, Radoslaw M Biernacki, Shao Hua Chen, Piotr A Grobelny, and Ronald H Hemmers. Space mapping technique for electromagnetic optimization. *IEEE Transactions on microwave theory and techniques*, 42(12):2536–2544, 1994.

[13] Anand Amrit, Leifur Leifsson, and Slawomir Koziel. Fast multi-objective aerodynamic optimization using sequential domain patching and multifidelity models. *Journal of Aircraft*, 57(3):388–398, 2020.

[14] Slawomir Koziel and Leifur Leifsson. Multi-level cfd-based airfoil shape optimization with automated low-fidelity model selection. *Procedia Computer Science*, 18:889–898, 2013.

[15] Leifur Leifsson and Slawomir Koziel. Aerodynamic shape optimization by variable-fidelity computational fluid dynamics models: a review of recent progress. *Journal of Computational Science*, 10:45–54, 2015.

[16] Marc C Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[17] John McFarland and Sankaran Mahadevan. Multivariate significance testing and model calibration under uncertainty. *Computer methods in applied mechanics and engineering*, 197(29-32):2467–2479, 2008.

[18] Matthew Plumlee. Bayesian calibration of inexact computer models. *Journal of the American Statistical Association*, 112(519):1274–1285, 2017.

[19] Dave Higdon, Marc Kennedy, James C Cavendish, John A Cafeo, and Robert D Ryne. Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing*, 26(2):448–466, 2004.

[20] Daniel W Apley, Jun Liu, and Wei Chen. Understanding the effects of model uncertainty in robust design with computer experiments. 2006.

[21] Maria J Bayarri, James O Berger, Rui Paulo, Jerry Sacks, John A Cafeo, James Cavendish, Chin-Hsu Lin, and Jian Tu. A framework for validation of computer models. *Technometrics*, 49(2):138–154, 2007.

[22] Paul D Arendt, Daniel W Apley, Wei Chen, David Lamb, and David Gorsich. Improving identifiability in model calibration using multiple responses. 2012.

[23] Paul D Arendt, Daniel W Apley, and Wei Chen. Quantification of model uncertainty: Calibration, model discrepancy, and identifiability. 2012.

[24] David A Stainforth, Tolu Aina, Carl Christensen, Mat Collins, Nick Faull, Dave J Frame, Jamie A Kettleborough, S Knight, A Martin, JM Murphy, et al. Uncertainty in predictions of the climate response to rising levels of greenhouse gases. *Nature*, 433(7024):403–406, 2005.

[25] Weizhao Zhang, Ramin Bostanabad, Biao Liang, Xuming Su, Danielle Zeng, Miguel A Bessa, Yanchao Wang, Wei Chen, and Jian Cao. A numerical bayesian-calibrated characterization method for multiscale prepreg preforming simulations with tension-shear coupling. *Composites Science and Technology*, 170:15–24, 2019.

[26] Robert B Gramacy, Derek Bingham, James Paul Holloway, Michael J Grosskopf, Carolyn C Kuranz, Erica Rutter, Matt Trantham, and R Paul Drake. Calibrating a large computer experiment simulating radiative shock hydrodynamics. *The Annals of Applied Statistics*, 9(3):1141–1168, 2015.

[27] Lluis Jofre, Gianluca Geraci, Hillary Fairbanks, Alireza Doostan, and Gianluca Iaccarino. Multi-fidelity uncertainty quantification of irradiated particle-laden turbulence. *arXiv preprint arXiv:1801.06062*, 2018.

[28] Mfnets: data efficient all-at-once learning of multifidelity surrogates as directed networks of information sources. *Computational Mechanics*, 68(4):741–758, 2021.

[29] Alex A Gorodetsky, John D Jakeman, Gianluca Geraci, and Michael S Eldred. Mfnets: multi-fidelity data-driven networks for bayesian learning and prediction. *International Journal for Uncertainty Quantification*, 10(6), 2020.

[30] Rebecca E Morrison, Todd A Oliver, and Robert D Moser. Representing model inadequacy: A stochastic operator approach. *SIAM/ASA Journal on Uncertainty Quantification*, 6(2):457–496, 2018.

[31] Rebecca E Morrison. Embedded discrepancy operators in reduced models of interacting species. *arXiv preprint arXiv:1910.08191*, 2019.

[32] Teresa Portone, Damon McDougall, and Robert D Moser. A stochastic operator approach to model inadequacy with applications to contaminant transport. *arXiv preprint arXiv:1702.07779*, 2017.

[33] Jonathan Tammer Eweis-Labolle, Nicholas Oune, and Ramin Bostanabad. Data fusion with latent map gaussian processes. *Journal of Mechanical Design*, 144(9):091703, 2022.

[34] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[35] Liang Yan and Tao Zhou. An adaptive surrogate modeling based on deep neural networks for large-scale bayesian inverse problems. 11 2019.

[36] Xuhui Meng and George Em Karniadakis. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse pde problems. *Journal of Computational Physics*, 401:109020, 2020.

[37] Subhayan De, Jolene Britton, Matthew Reynolds, Ryan Skinner, Kenneth Jansen, and Alireza Doostan. On transfer learning of neural networks using bi-fidelity data for uncertainty propagation. *International Journal for Uncertainty Quantification*, 10(6), 2020.

[38] Suraj Pawar, Omer San, Prakash Vedula, Adil Rasheed, and Trond Kvamsdal. Multi-fidelity information fusion with concatenated neural networks. *Scientific Reports*, 12(1):5900, Apr 2022.

[39] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[40] Nicholas Oune and Ramin Bostanabad. Latent map gaussian processes for mixed variable metamodeling. *Computer Methods in Applied Mechanics and Engineering*, 387:114128, 2021.

[41] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[42] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[43] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

[44] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[45] John Mitros and Brian Mac Namee. On the validity of bayesian neural networks for uncertainty estimation. *arXiv preprint arXiv:1912.01530*, 2019.

[46] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International conference on machine learning*, pages 5436–5446. PMLR, 2020.

[47] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

[48] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[49] Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2):29–48, 2022.

[50] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–6, 2000.

[51] D. L. Donoho and C. Grimes. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. *Proc Natl Acad Sci U S A*, 100(10):5591–6, 2003.

[52] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–23, 2000.

[53] Ashutosh Saxena, Abhinav Gupta, and Amitabha Mukerjee. Non-linear dimensionality reduction by locally linear isomaps. In *Neural Information Processing*, pages 1038–1043. Springer.

[54] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[55] N. Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research*, 6(Nov):1783–1816, 2005.

[56] Francois Chollet. *Deep learning with python*. Manning Publications Co., 2017.

[57] Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[58] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

[59] Meire Fortunato, Charles Blundell, and Oriol Vinyals. Revisiting bayes by backprop. 2018.

[60] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

[61] Kurt Cutajar, Mark Pullin, Andreas Damianou, Neil Lawrence, and Javier González. Deep gaussian processes for multi-fidelity modeling. *arXiv preprint arXiv:1903.07320*, 2019.

[62] Lixue Liu, Xueguan Song, Chao Zhang, and Dacheng Tao. Gan-mdf: An enabling method for multifidelity data fusion. *IEEE Internet of Things Journal*, 9(15):13405–13415, 2022.

[63] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1050–1059. JMLR.org, 2016.

[64] George J Dvorak. Transformation field analysis of inelastic composite materials. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 437(1900):311–327, 1992.

[65] Jean-Claude Michel and Pierre Suquet. Nonuniform transformation field analysis. *International journal of solids and structures*, 40(25):6937–6955, 2003.

[66] Sophie Roussette, Jean-Claude Michel, and Pierre Suquet. Nonuniform transformation field analysis of elastic–viscoplastic composites. *Composites Science and Technology*, 69(1):22–27, 2009.

[67] Zeliang Liu, MA Bessa, and Wing Kam Liu. Self-consistent clustering analysis: an efficient multi-scale scheme for inelastic heterogeneous materials. *Computer Methods in Applied Mechanics and Engineering*, 306:319–341, 2016.

[68] Gengdong Cheng, Xikui Li, Yinghao Nie, and Hengyang Li. Fem-cluster based reduction method for efficient numerical prediction of effective properties of heterogeneous material in nonlinear range. *Computer Methods in Applied Mechanics and Engineering*, 348:157–184, 2019.

[69] Shiguang Deng, Carl Soderhjelm, Diran Apelian, and Ramin Bostanabad. Reduced-order multiscale modeling of plastic deformations in 3d alloys with spatially varying porosity by deflated clustering analysis. *Computational Mechanics*, 70(3):517–548, 2022.

[70] Zeliang Liu, Mark Fleming, and Wing Kam Liu. Microstructural material database for self-consistent clustering analysis of elastoplastic strain softening materials. *Computer Methods in Applied Mechanics and Engineering*, 330:547–577, 2018.

[71] Christian Miehe. Numerical computation of algorithmic (consistent) tangent moduli in large-strain computational inelasticity. *Computer methods in applied mechanics and engineering*, 134(3-4):223–240, 1996.

[72] Varvara Kouznetsova, WAM Brekelmans, and FPT1005 Baaijens. An approach to micro-macro modeling of heterogeneous materials. *Computational mechanics*, 27(1):37–48, 2001.

[73] Shaoqiang Tang, Lei Zhang, and Wing Kam Liu. From virtual clustering analysis to self-consistent clustering analysis: a mathematical study. *Computational Mechanics*, 62:1443–1460, 2018.

[74] Marcel R Ackermann, Johannes Blömer, Daniel Kuntze, and Christian Sohler. Analysis of agglomerative clustering. *Algorithmica*, 69:184–215, 2014.

[75] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.

[76] Gui-Rong Liu. *Meshfree methods: moving beyond the finite element method*. CRC press, 2009.

[77] Praveen Yadav and Krishnan Suresh. Large scale finite element analysis via assembly-free deflated conjugate gradient. *Journal of Computing and Information Science in Engineering*, 14(4), 2014.

[78] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*, pages 63–71. Springer, 2004.

[79] Ramin Bostanabad, Yu-Chin Chan, Liwei Wang, Ping Zhu, and Wei Chen. Globally approximate gaussian processes for big data with application to data-driven metamaterials design. *Journal of Mechanical Design*, 141(11), 2019.

[80] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[81] Ramin Bostanabad, Yichi Zhang, Xiaolin Li, Tucker Kearney, L Catherine Brinson, Daniel W Apley, Wing Kam Liu, and Wei Chen. Computational microstructure characterization and reconstruction: Review of the state-of-the-art techniques. *Progress in Materials Science*, 95:1–41, 2018.

[82] Shiguang Deng, Carl Soderhjelm, Diran Apelian, and Krishnan Suresh. Estimation of elastic behaviors of metal components containing process induced porosity. *Computers & Structures*, 254:106558, 2021.

[83] Shiguang Deng, Diran Apelian, and Ramin Bostanabad. Concurrent multiscale damage analysis with adaptive spatiotemporal dimension reduction. *arXiv preprint arXiv:2205.12149*, 2022.

[84] Shiguang Deng, Carl Soderhjelm, Diran Apelian, and Krishnan Suresh. Second-order defeaturing estimator of manufacturing-induced porosity on structural elasticity. *International Journal for Numerical Methods in Engineering*, 123(19):4483–4517, 2022.

[85] Byeng D Youn, Byung C Jung, Zhimin Xi, Sang Bum Kim, and WR Lee. A hierarchical framework for statistical model calibration in engineering product development. *Computer Methods in Applied Mechanics and Engineering*, 200(13-16):1421–1431, 2011.

[86] Alaa Olleak and Zhimin Xi. Calibration and validation framework for selective laser melting process based on multi-fidelity models and limited experiment data. *Journal of Mechanical Design*, 142(8):081701, 2020.

[87] Ramin Bostanabad, Biao Liang, Jiaying Gao, Wing Kam Liu, Jian Cao, Danielle Zeng, Xuming Su, Hongyi Xu, Yang Li, and Wei Chen. Uncertainty quantification in multiscale simulation of woven fiber composites. *Computer Methods in Applied Mechanics and Engineering*, 338:506–532, 2018.

[88] Henry C. Herbol, Weici Hu, Peter Frazier, Paulette Clancy, and Matthias Poloczek. Efficient search of compositional space for hybrid organic–inorganic perovskites via bayesian optimization. *npj Computational Materials*, 4(1), 2018.

Appendix A

Probabilistic Neural Data Fusion for Learning from an Arbitrary Number of Multi-fidelity Data Sets

We provide the formulations of the analytic problems in Appendix A.1, the background and details of the real-world problems in Appendix A.2, and the methodology and details of the FFNN and SMF methods in Appendix A.3.

A.1 Table of Analytic Examples

Table A.2 details the analytic functions used for the examples covered in Section 2.4. For each multi-fidelity problem, we calculate the accuracy of each LF source with respect to the HF source via normalized root mean squared error (NRMSE):

$$\text{NRMSE} = \sqrt{\frac{(\boldsymbol{y}^l - \boldsymbol{y}^h)^T(\boldsymbol{y}^l - \boldsymbol{y}^h)}{10000 \times \text{var}(\boldsymbol{y}^h)}} \tag{A.1-1}$$

where $\boldsymbol{y}^l$ and $\boldsymbol{y}^h$ are 10000×1 arrays of outputs sampled randomly via Sobol sequence from the LF and HF sources, respectively. We use the same sample locations and outputs as our test data when evaluating MSE and IS in Section 2.4.1 and Section 2.4.2.

A.2 Background on Real-World Examples

The DNS-ROM data set is extracted from the work presented in Chapter 3 [2] and is detailed in Section 3.5.2. In this work, the goal is to accelerate multiscale damage simulations of cast aluminum alloys by replacing direction numerical simulations (DNS) that are done at the microscale via reduced-order models (ROMs). These ROMs provide computational acceleration by solving a reduced-order representation of the governing equations and, depending on how much the governing equations are simplified (to achieve speedups), provide different levels of accuracy with respect to the DNS. A data point in the DNS-ROM data set contains the toughness of a microstructure as a function of four microstructural descriptors (pore volume fraction, number of pores, pore aspect ratio, average nearest neighbor distance among the pores) and two material properties which affect the damage behavior of the microstructure (evolutionary rate parameter and critical effective plastic strain). Hence, this data set has 1 output[1] (toughness) and 6 inputs. The response of each sample is obtained via either DNS (HF source) or one of three ROMs (LF sources) which differ in terms of their accuracy and cost (i.e., there is a total of four sources). The data set has $n^h = 70, n^{l_1} = 110, n^{l_2} = 170, n^{l_3} = 250$ samples.

The HOIP problem deals with the composition of HOIP crystals, which are a relatively recently developed class of materials with desirable photovoltaic properties for applications in solar cells [88]. The quantity of interest is the inter-molecular binding energy between a HOIP and solvent pair used in a solar cell, which is used as an indicator of the cell's photovoltaic

[1]Note that we only consider one of the responses shown in Table 3.5 (i.e., toughness) for the experiments performed in Section 2.4.3.

conversion efficiency. HOIPs are perovskite materials and therefore their crystals have an ABX_3 configuration, i.e., they are composed of three compounds corresponding to sites A (organic or inorganic cation), B (inorganic metal cation), and X_3 (halide combination). There are a number of choices for each site and any combination of these choices is feasible, so the possible HOIP compositions comprise a very large combinatorial space. The data set we use in this paper has three categorical inputs with $l_1 = 10$, $l_2 = 3$, and $l_3 = 16$ levels which correspond to 10 possible X_3 halide compositions[2], 3 possible choices for the organic/inorganic A site cation[3], and 16 possible choices of solvent[4], respectively. The B site cation is held constant as lead, as other choices are exceedingly rare [88]. The output is the inter-molecular binding energy between the input HOIP and solvent. There are one HF and three LF data sets with unknown levels of fidelity and $n^h = 480, n^{l_1} = 480, n^{l_2} = 179, n^{l_3} = 240$ samples. We use 90% of the available samples for each source for training and 10% for testing. We refer the reader to [88] for further details on HOIPs.

A.3 Other Multi-Fidelity NN-Based Approaches

A.3.1 Feedforward Neural Networks

As depicted in Figure A.1, for MF modeling via an FFNN we simply feed the numerical inputs $\boldsymbol{x}$, the prior representation of the source indicator $\boldsymbol{\zeta}(t^s)$ and the prior representation of the categorical inputs $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ into the FFNN to produce the output. This approach has two clear disadvantages with respect to Pro-NDF: (1) it does not provide a tool such as the fidelity manifold of Pro-NDF that provides a direct visualization of the correlation between the data sources, and (2) it has a fully deterministic setting which does not enable uncertainty

[2]All possible three-element combinations of chlorine, bromine, and iodine.
[3]Methylammonium, formamidinium, or cesium.
[4]Acetone, methanol, chloroform, etc.

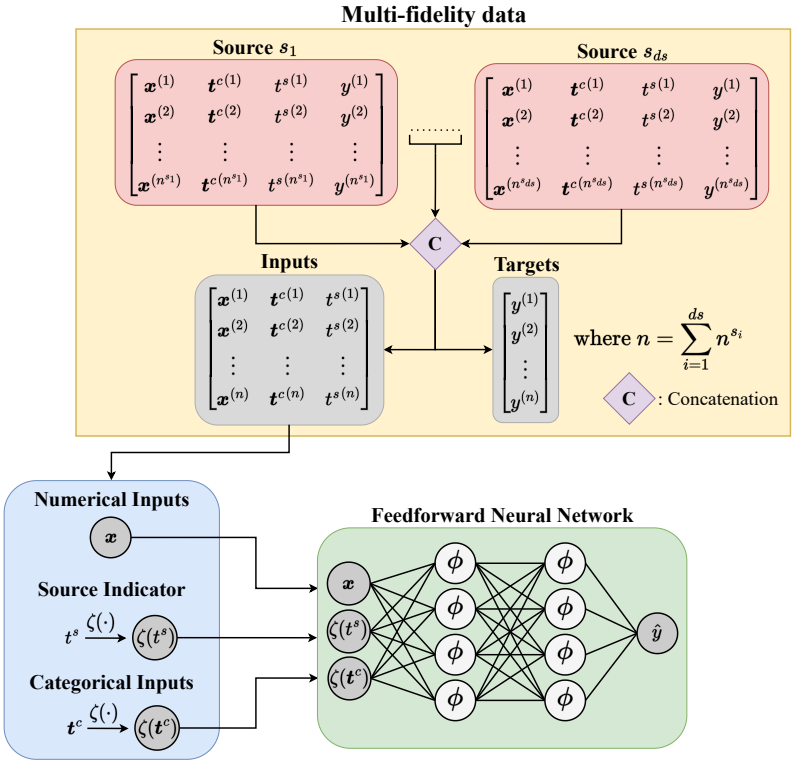

Figure A.1 FFNN for multi-fidelity modeling: As Pro-NDF , this approach allows to use an arbitrary number of sources by appending a source indicator variable to each data set and concatenating them. The FFNN maps the numerical inputs $\boldsymbol{x}$, a priori representation of the source indicator $\boldsymbol{\zeta}(t^s)$, and categorical inputs $\boldsymbol{\zeta}(\boldsymbol{t}^c)$ to the output.

quantification and thus using a loss function based on proper scoring rules. In particular, we use the following loss function for training the FFNN:

$$\mathcal{L} = \mathcal{L}_{MSE} + \beta \mathcal{L}_2 \tag{A.3-2}$$

where $\mathcal{L}_{MSE}$ is the mean squared error of the predictions and $\mathcal{L}_2$ is $L2$ regularization:

$$\mathcal{L}_{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2 \tag{A.3-3}$$

$$\mathcal{L}_2 = |\boldsymbol{\theta}|^2 \tag{A.3-4}$$

We employ Adam as the optimizer and use RayTune [60] and Hyperopt with five-fold cross-validation to find the optimum architecture and hyperparameters which include the learning rate, regularization parameter β, and batch size N. For further details on implementation, please see our GitLab repository.

A.3.2 Sequential Multi-Fidelity Networks

SMF is our extension of the approach detailed in [35] to more than two sources of data. Unlike the other methods presented in this paper, multi-fidelity modeling via SMF requires training a separate surrogate for each data source. As depicted in Figure A.2, individual FFNNs are trained for each source in the sequence that ends with the HF source. After a surrogate is trained for a data source, its outputs are used to augment the inputs of the next model in the sequence and hence the resulting input-output relationships are:

$$\begin{aligned}
\hat{y}^{s_1} &= \hat{f}^{s_1}\left(\boldsymbol{u}^{s_1}\right) \\
\hat{y}^{s_2} &= \hat{f}^{s_2}\left(\boldsymbol{u}^{s_2}, \hat{y}^{s_1}(\boldsymbol{u}^{s_2})\right) \\
&\cdots \\
\hat{y}^{s_{ds}} &= \hat{f}^{s_{ds}}\left(\boldsymbol{u}^{s_{ds}}, \hat{y}^{s_{ds-1}}(\boldsymbol{u}^{s_{ds}})\right)
\end{aligned} \tag{A.3-5}$$

where $\hat{y}^{s_i}$ is the output of the FFNN , $\hat{f}^{s_i}$ is the mapping defined by the FFNN, $\boldsymbol{u}^{s_i}$ is the combined numeric and categorical input $\boldsymbol{u} = [\boldsymbol{x}, \boldsymbol{\zeta}(\boldsymbol{t}^c)]$, and i denotes the data source with $i = ds$ being the HF source. Each individual FFNN employs the same loss function and optimizer as in the FFNN method presented in Appendix A.3.1.

Like MF-DGP, the SMF approach is highly sensitive to the ordering of the data sources in the sequence. In the case that the fidelity levels are known, they are assigned in the order of increasing fidelity, i.e., source 1 is the least accurate LF source while source $ds - 1$ is

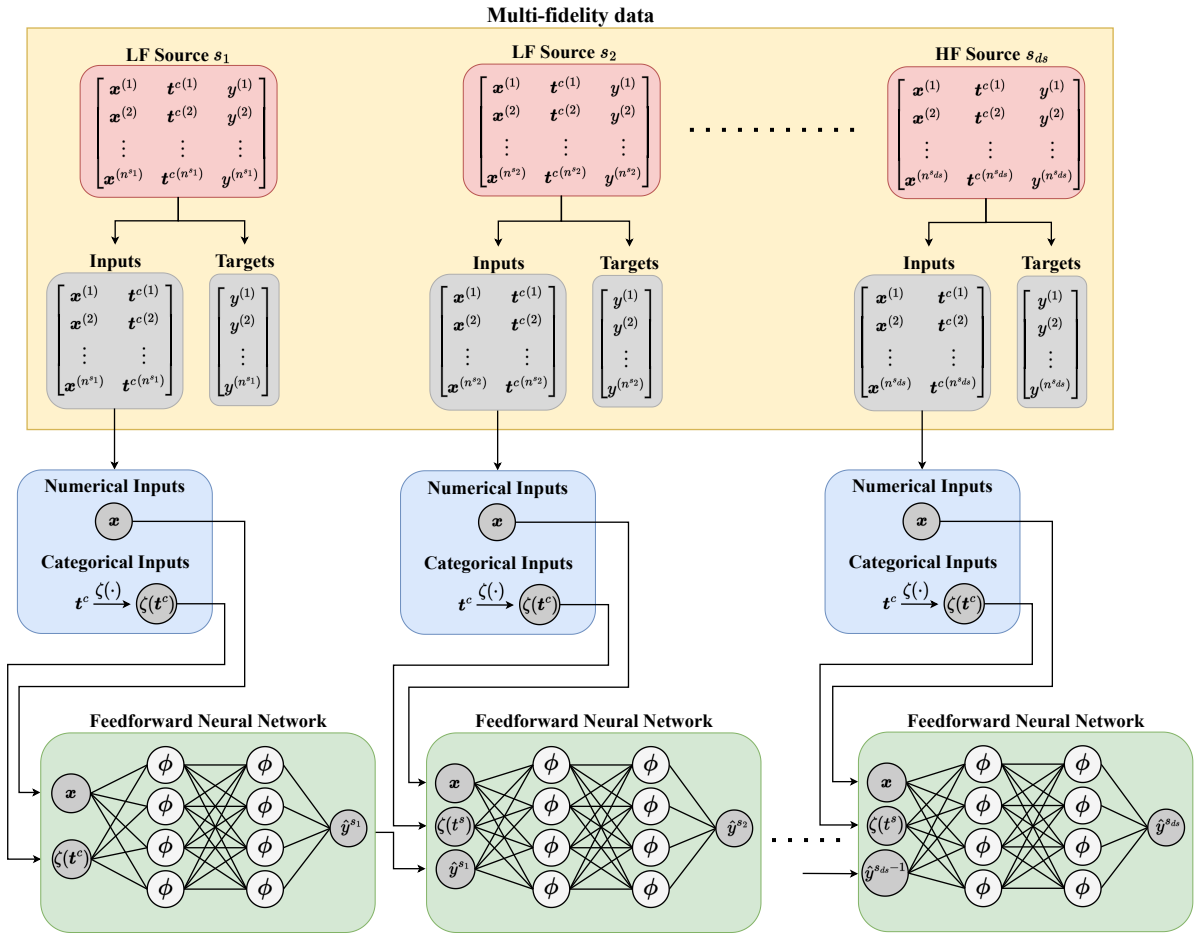

Figure A.2 Sequential Multi-fidelity (SMF) Networks: SMF is a hierarchical approach that relies on sequentially training a model (e.g., an FFNN) for each data source in an ascending order based on the fidelities. The inputs of a model are augmented with the outputs of the previous one until reaching the model of the HF source.

the most accurate. With this ordering, the SMF approach leverages the entire data set to achieve good HF prediction accuracy by minimizing the complexity of the mapping learned by each successive FFNN. However, in the case that the fidelities are not known, the order of the LF sources is assigned randomly. In this case, the mappings of the successive FFNNs no longer monotonically approaches that of the HF function, and the SMF approach is unable to properly leverage the additional LF data. In this paper, we assume that the fidelity levels are unknown and therefore assign the data source ordering randomly when using SMF.

Similar to the FFNN approach, the SMF approach does not provide a latent mapping and is

entirely deterministic. Like all hierarchical approaches, it also requires knowledge of fidelity levels for good performance. These factors lead to a marked disadvantage in the context of the problems examined in this paper, and we therefore expect the SMF method to perform poorly.

We use RayTune and Hyperopt with five-fold cross-validation to find the optimum architecture and hyperparameters for *each* FFNN in the SMF method. Namely, we tune the learning rate, regularization parameter β, and batch size N. We also tune an additional parameter that determines whether to use the numeric and categorical inputs $\boldsymbol{u}$ in the final FFNN, since the mapping may be simple enough to learn from just the previous FFNN outputs in the case that the last LF source is highly accurate. For further details on implementation, please see our GitLab repository.

A.4 Additional details on simulations

A.4.1 Computational time

The required training time for each MF approach and problem assessed in Section 2.4.2 and Section 2.4.3 is shown in Table A.1.

	Rational	Wing-weight	Borehole	DNS-ROM	HOIP
Pro-NDF	195.80	65.99	91.32	35.96	32.43
LMGP	3.35	49.40	133.88	127.53	112.95
MF-DGP[a,b]	304.12	983.16	2312.01	3533.04	-
SF-GP	6.43	5.80	4.88	5.10	12.76
FFNN	75.42	418.39	71.11	183.94	387.55
SMF	96.76	108.99	476.85	165.05	935.42

Table A.1 Training time for different models on each example (in seconds).

[a]Simulations for MF-DGP were conducted on a workstation equipped with an 11th Gen Intel Core i7-11700K CPU, with a clock speed of 3.60 GHz, 8 cores and 16 threads.

[b]MF-DGP cannot handle categorical variables and hence is not applied to the HOIP example.

Name	Source ID	Formulation	n	σ^2	NRMSE
Rational	$y^h(x)$	$\frac{1}{0.1x^3+x^2+x+1}$	5	0.001	-
	$y^{l_1}(x)$	$\frac{1}{0.2x^3+x^2+x+1}$	30	0.001	0.23
	$y^{l_2}(x)$	$\frac{1}{0\times x^3+x^2+x+1}$	30	0.001	0.15
	$y^{l_3}(x)$	$\frac{1}{0\times x^3+x^2+0\times x+1}$	30	0.001	0.73
Polynomial[a]	$y^h(\boldsymbol{x})$	$4x_1^2-2.1x_1^4+\frac{x_1^6}{3}-4x_2^2+4x_2^4+x_1x_2$	5	1	-
	$y^l(\boldsymbol{x})$	$y^h(0.7x_1,0.7x_2)+x_1x_2-65$	20	1	5.77
Wing-weight	$y^h(\boldsymbol{x})$	$0.036S_\omega^{0.758}W_{f\omega}^{0.0035}\left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6}q^{0.006}\times\lambda^{0.04}\left(\frac{100t_c}{\cos(\Lambda)}\right)^{-0.3}+(N_zW_{dg})^{0.49}+S_\omega W_p$	15	25	-
	$y^{l_1}(\boldsymbol{x})$	$0.036S_\omega^{0.758}W_{f\omega}^{0.0035}\left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6}q^{0.006}\times\lambda^{0.04}\left(\frac{100t_c}{\cos(\Lambda)}\right)^{-0.3}+(N_zW_{dg})^{0.49}+1\times W_p$	50	25	0.20
	$y^{l_2}(\boldsymbol{x})$	$0.036S_\omega^{0.8}W_{f\omega}^{0.0035}\left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6}q^{0.006}\times\lambda^{0.04}\left(\frac{100t_c}{\cos(\Lambda)}\right)^{-0.3}+(N_zW_{dg})^{0.49}+1\times W_p$	50	25	1.14
	$y^{l_3}(\boldsymbol{x})$	$0.036S_\omega^{0.9}W_{f\omega}^{0.0035}\left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6}q^{0.006}\times\lambda^{0.04}\left(\frac{100t_c}{\cos(\Lambda)}\right)^{-0.3}+(N_zW_{dg})^{0.49}+0\times W_p$	50	25	5.75
Borehole	$y^h(\boldsymbol{x})$	$\frac{2\pi T_u(H_u-H_l)}{\ln\left(\frac{r}{r_w}\right)\left(1+\frac{2LT_u}{\ln\left(\frac{r}{r_w}\right)r_w^2k_w}+\frac{T_u}{T_l}\right)}$	15	6.25	-
	$y^{l_1}(\boldsymbol{x})$	$\frac{2\pi T_u(H_u-0.8H_l)}{\ln\left(\frac{r}{r_w}\right)\left(1+\frac{1LT_u}{\ln\left(\frac{r}{r_w}\right)r_w^2k_w}+\frac{T_u}{T_l}\right)}$	50	6.25	3.67
	$y^{l_2}(\boldsymbol{x})$	$\frac{2\pi T_u(H_u-3H_l)}{\ln\left(\frac{r}{r_w}\right)\left(1+\frac{8LT_u}{\ln\left(\frac{r}{r_w}\right)r_w^2k_w}+0.75\frac{T_u}{T_l}\right)}$	50	6.25	3.73
	$y^{l_3}(\boldsymbol{x})$	$\frac{2\pi T_u(1.1H_u-H_l)}{\ln\left(\frac{4r}{r_w}\right)\left(1+\frac{3LT_u}{\ln\left(\frac{r}{r_w}\right)r_w^2k_w}+\frac{T_u}{T_l}\right)}$	50	6.25	0.38
	$y^{l_4}(\boldsymbol{x})$	$\frac{2\pi T_u(1.05H_u-H_l)}{\ln\left(\frac{2r}{r_w}\right)\left(1+\frac{2LT_u}{\ln\left(\frac{r}{r_w}\right)r_w^2k_w}+\frac{T_u}{T_l}\right)}$	50	6.25	0.19

Table A.2 Table of analytic functions: The analytic examples have different input dimensionality, number of sources, and forms of model error. n denotes the number of samples, σ^2 is the variance of the noise, and NRMSE is the normalized root mean squared error of an LF source with respect to an HF source, see Equation (A.1-1).

[a]Extracted from [62].