

PHYSICS-INFORMED MACHINE LEARNING FOR UNCERTAINTY QUANTIFICATION
AND OPTIMIZATION

By

Berkcan Kapusuzoglu

Dissertation

Submitted to the Faculty of the
Graduate School of Vanderbilt University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in

Civil Engineering

May 13, 2022

Nashville, Tennessee

Approved:

Sankaran Mahadevan, Ph.D.

Douglas E. Adams, Ph.D.

Pranav M. Karve, Ph.D.

D. Mitchell Wilkes, Ph.D.

Copyright © 2022 Berkcan Kapusuzoglu
All Rights Reserved

To my wife, Hannah, and my daughter Luna Rey

ACKNOWLEDGMENTS

I would like to express my grateful appreciation to my adviser, Professor Sankaran Mahadevan for his guidance, continuous encouragement and work ethic during the course of this study. He believed in me and reflected his joy and passion he has for his research to me. I would also like to thank Professor Douglas Adams, Dr. Pranav Karve and Professor Mitchell Wilkes for their valuable recommendations. I am also thankful to CEE staff for their support and Matthew Sato for his patience in conducting the experiments at Laboratory for Systems Integrity and Reliability.

My sincerest thanks and gratitude are extended to my parents, Filiz and Mehmet, my grandparents, Nuran and Yilmaz, my aunt Selda, my uncles, Ali and Hakan, and finally to my wife Hannah for their infinite love and continuous support. They have given me everything I needed throughout my life, and mere words cannot express my appreciation. I am deeply grateful for their sacrifices on my behalf, as I have strived to excel at being the best version of myself.

I am also grateful to work with so many brilliant people at Vanderbilt University. In particular, the valuable discussions with Risk, Reliability and Resilience (RRR) research group: Dr. Abhinav Subramanian, Dr. Paromita Nath, Dr. Xiaoge Zhang, Dr. Yanqing Bao, Dr. Kyle Neal, Sarah Miele, Yulin Guo, William Sisson, Yingxiao Kong, Andrew White, and Oliver Stover.

This research was partly supported by a grant from the National Institute of Standards and Technology under the Smart Manufacturing Data Analytics Project (Cooperative Agreement No. 70NANB14H036 and 70NANB18H245) and Mitsubishi Heavy Industries in Nagasaki and Takasago, Japan.

TABLE OF CONTENTS

	Page
DEDICATION	iii
ACKNOWLEDGMENTS	iv
LIST OF TABLES	x
LIST OF FIGURES	xi
1 Introduction	1
1.1 Overview	1
1.2 Research Objectives	4
1.3 Organization of the Dissertation	5
2 Background	8
2.1 Dimension Reduction	8
2.1.1 Singular Value Decomposition (SVD)	8
2.1.2 Randomized SVD (rSVD)	9
2.2 Machine Learning (ML)	11
2.2.1 Gaussian Process (GP) Surrogate Modeling	11
2.2.2 Deep Neural Network (DNN)	12
2.2.3 Bayesian Neural Network (BNN)	13
2.3 Uncertainty Quantification	14
2.3.1 Bayesian Calibration	16
2.3.2 Particle Filter	16
2.3.3 Bayesian Networks	19
2.4 Global Sensitivity Analysis (GSA)	19
2.5 Fused Filament Fabrication (FFF)	20
2.5.1 FFF Temperature Modeling	21
2.5.2 Bond Formation Modeling	23
2.6 Summary	26
3 Machine Learning using both Physics Knowledge and Experimental Data	28
3.1 Introduction	28
3.2 Proposed Methodology	30
3.2.1 Physics-Informed Machine Learning (PIML) for Additive Manufacturing	30

3.2.1.1	Physics-Informed Loss Functions	31
3.2.1.2	Physics Model Output as Additional ML Model Input	33
3.2.1.3	Pre-trained PIML Model	33
3.2.1.4	Combination of PIML Strategies	34
3.3	Numerical Example	36
3.3.1	Problem Setup	36
3.3.2	Model Training and Prediction	37
3.3.3	Model Performance	37
3.4	Summary	40
4	Sensitivity Analysis with Hybrid Machine Learning Models	42
4.1	Introduction	42
4.2	Proposed Methodology	45
4.2.1	Implementation of PIML Strategies in ML Models	47
4.2.1.1	Implementation of PIML in GP Models	47
4.2.2	Variance of GP and DNN Prediction	50
4.2.3	Sobol' Indices Computation with Model Uncertainty	51
4.3	Numerical Example	53
4.3.1	Training Details of the ML Models	53
4.3.2	Comparison of Computational Effort	55
4.3.3	Comparison of Accuracy	56
4.3.4	GSA Results using GP Models	59
4.3.5	GSA Results using DNN Models with MC Dropout	60
4.4	Summary	63
5	Adaptive Surrogate Modeling for High-Dimensional Spatio-Temporal Output	64
5.1	Introduction	64
5.2	Proposed Methodology	67
5.2.1	Dimension Reduction	68
5.2.2	Surrogate Model Construction	69
5.2.3	Surrogate Model Error Quantification	70
5.2.4	Proposed Adaptive Sample Selection Method	71
5.2.4.1	Space-Filling Criterion	72
5.2.4.2	Expected Improvement	72

5.2.4.3	Proposed Learning Function	73
5.2.4.4	Stopping Criteria	74
5.3	Summary of Methodology	75
5.4	Evaluation of Proposed Approach using Benchmark Problems	75
5.5	Engineering Application	80
5.5.1	Generation of Initial Training Data	81
5.5.2	Dimension Reduction of the Output Space	81
5.5.3	Surrogate Model Construction and Cross-Validation	82
5.5.4	Adaptive Surrogate Modeling	83
5.6	Summary	87
6	Multi-Level Information Fusion for Model Calibration	88
6.1	Introduction	88
6.2	Proposed Methodology	90
6.2.1	Bayesian Calibration with Multiple Types of Data	90
6.2.2	Offline and Online Multi-Level Bayesian Calibration	93
6.3	Numerical Example	95
6.3.1	Introduction to the Problem	95
6.3.2	Bayesian Parameter Calibration	96
6.4	Summary	105
7	Decision-Making under Uncertainty	107
7.1	Introduction	107
7.2	Single Physical Quantity of Interest	109
7.2.1	Proposed Methodology	111
7.2.1.1	Uncertainty Quantification in FFF	111
7.2.1.1.1	Model Calibration under Uncertainty	112
7.2.1.2	Process Optimization under Uncertainty	114
7.2.1.3	Experimental Work	116
7.2.1.4	Numerical Example	117
7.2.1.4.1	Prediction of the Cooling of Filaments	118
7.2.1.4.2	Contribution Assessment of Uncertainty Sources	120
7.2.1.4.2.1	GSA of the Bond Formation Model	121
7.2.1.4.3	Model Calibration	124

7.2.1.4.4	Surrogate Modeling	125
7.2.1.4.5	Process Design Optimization	127
7.2.1.5	Summary	131
7.3	Multiple Quantities of Interest	132
7.3.1	Proposed Methodology	132
7.3.1.1	Data Collection	132
7.3.1.2	Construction of Data-Driven Prediction Model	133
7.3.1.2.1	Uncertainty Quantification in Bayesian Neural Network (BNN)	133
7.3.1.3	Multi-Objective Optimization under Uncertainty	135
7.3.1.3.1	Formulation of Multi-Objective Optimization	135
7.3.1.3.2	Solution of Multi-Objective Optimization	136
7.3.1.3.2.1	Non-Dominated Sorting Genetic Algorithm II	136
7.3.1.3.2.2	Evaluating Performance: Hypervolume Indicator	137
7.3.1.3.3	Experimental Validation	138
7.3.1.3.3.1	BNN Model Validation	138
7.3.1.3.3.2	Optimization Solution Validation	138
7.3.1.4	Summary of Methodology	139
7.3.2	Numerical Example	139
7.3.2.1	Data Collection from Experiments	140
7.3.2.2	Model Training and Prediction	142
7.3.2.3	Model Performance and Cross-Validation	143
7.3.2.4	Multi-Objective Optimization	145
7.3.2.4.1	Case 1	148
7.3.2.4.2	Case 2	149
7.3.2.4.3	Case 3	150
7.3.2.4.4	Case 4	150
7.3.2.5	Monte Carlo Optimization	153
7.3.2.6	Experimental Validation	155
7.3.2.7	Summary	156
8	Conclusion	158
8.1	Summary of Contributions	158

8.2 Future Work	159
A Appendix	161
A.1 Benchmark Test Functions	161
REFERENCES	162

LIST OF TABLES

Table		Page
3.1	Effect of different amounts of training data on the RMSE of different ML models	39
4.1	Computational effort of eight models for training and estimation of first-order and total effect Sobol' indices using 5000 MC samples with $n = 39$ number of observations	56
4.2	Effect of different amounts of training data on the RMSE of the GP models . . .	57
4.3	Effect of different amounts of training data on the RMSE of the DNN models . .	57
4.4	Tenfold cross-validation average RMSE results of GP and DNN models	57
4.5	First-order sensitivity estimate prediction bounds of nozzle temperature for different amounts of experimental training data, using the GP models	60
4.6	First-order sensitivity estimate prediction bounds of nozzle temperature for different amounts of experimental training data, using the DNN models	62
5.1	Physics model input and output parameters	81
5.2	7-fold cross-validation results in terms of RMSE	82
5.3	Average leave-one-out cross-validation (LOOCV) results in PNMAE, PNrMAE, and RMSE	87
6.1	Surrogate model input parameters and ranges	96
6.2	Experimental data	97
7.1	Process parameters and material properties	118
7.2	Intra-layer bond length measurements at each layer for $(T_n, v_p) = (240^\circ\text{C}, 0.042 \text{ m/s})$	125
7.3	Lower and upper bounds for the process design variables	128
7.4	Optimal printer nozzle temperatures ($^\circ\text{C}$) and extrusion velocities (m/s)	129
7.5	Overall average bond length at optimal solutions (all units are in millimeters) . .	130
7.6	Total void areas and overall mean bond lengths at $z_{\text{cut}} = 0.0175 \text{ m}$	131
7.7	Bond length measurements (in mm) at each layer for $(T_e, V_e, l_t) = (227^\circ\text{C}, 41 \text{ mm/s}, 0.6 \text{ mm})$	141
A.1	Test problems	161

LIST OF FIGURES

Figure	Page
2.1	A fully connected deep neural network (DNN) with two hidden layers 12
2.2	Relating model output to observation data 16
2.3	Schematic of FFF 21
2.4	Possible contact areas of a filament in the: (a) first layer and (b) remaining layers 22
2.5	Evolution of neck diameter during sintering process 23
2.6	Cross-sectional geometry of an FFF part printed with an extrusion temperature 240°C, speed 42 mm/s, initial layer height 0.7 mm, filament width 0.8 mm, filament length 35 mm, 6 number of layers and 15 filaments per layer 24
3.1	PIML strategies: (a) incorporate physics constraints within the loss function of the DNN, (b) use physics model outputs as additional inputs to the DNN model, and (c) pre-training a DNN model with physics model input-output and updating it with experimental data 35
3.2	Design of experiments for physics-based simulations and experiments 38
3.3	Performance and physical inconsistency of proposed models 39
3.4	Comparison of model prediction with test data: (a) for the first 4 models, and (b) for the last 4 models 40
4.1	PIML strategies: (a) incorporating physics-based loss functions in the ML models to enforce physics constraints, (b) pre-training an ML model with physics model input-output ($\mathbf{X}_{\text{phy}}, \mathbf{Y}_{\text{phy}}$) and updating it with experimental training data ($\mathbf{X}_{\text{obs}}, \mathbf{Y}_{\text{obs}}$), (c) the trained ML model prediction ($\hat{\mathbf{Y}}$), (d) updated ML model prediction ($\hat{\mathbf{Y}}$) 46
4.2	RMSE values of DNN models with varying dropout rates 58
4.3	First-order sensitivity index estimators for (a) nozzle temperature, and (b) nozzle speed, using two different pre-training and updating approaches for GP 58
4.4	Total effect sensitivity index estimates for (a) nozzle temperature, and (b) nozzle speed, using two different pre-training and updating approaches for GP 58
4.5	First-order sensitivity index estimates for (a) nozzle temperature, and (b) nozzle speed, using the GP models 59

4.6	Total effect sensitivity index estimates for (a) nozzle temperature, and (b) nozzle speed, using the GP models	60
4.7	First-order sensitivity index estimators for (a) nozzle temperature, and (b) nozzle speed, using DNN models with MC dropout	61
4.8	Total effect sensitivity index estimators for (a) nozzle temperature, and (b) nozzle speed, using DNN models with MC dropout	61
5.1	Schematic describing the proposed procedure for surrogate model construction with high-dimensional field data using rSVD	76
5.2	Evolution of the NRMSE from the initial surrogate model (20 samples) to the final surrogate model (45 samples) using different adaptive sampling techniques, for the seven benchmark functions in Appendix A.1	77
5.3	Adaptively selected training point locations for the Branin’s function and the final surrogate model accuracy in terms of NRMSE are plotted in contours (Black dots: initial 20 samples, Red dots: additional 25 samples)	78
5.4	Adaptively selected training point locations for the Schwefel function and the final surrogate model accuracy in terms of NRMSE are plotted in contours (Black dots: initial 20 samples, Red dots: additional 25 samples)	78
5.5	Predicted vs. observed values of important features for FEM run 23	83
5.6	Actual Feature 1 values vs. TBC thickness and T1T rate	83
5.7	Initial FEM training data and adaptively selected additional samples	85
5.8	Final surrogate model prediction accuracy in the original space for leave-one-out FEM runs	86
5.9	Final surrogate model prediction compared against FEM run 36 at node 5057	86
6.1	A simple hierarchical Bayesian network	91
6.2	A simple hierarchical dynamic Bayesian network	92
6.3	Flowchart describing the steps of particle filter algorithm	96
6.4	(a) Multi-level Bayesian network and (b) Offline calibration in multi-component system	98
6.5	Posterior distributions of model parameters using elongation data at the blade tip (blade ID 90), using offline Bayesian calibration	99

6.6	Posterior distributions of noise standard deviation and model discrepancy hyperparameters using elongation data at the blade tip (blade ID 90), using offline Bayesian calibration	100
6.7	Posterior distributions of model parameters using 3D deformation data of blade ID 01, using offline Bayesian calibration	100
6.8	Posterior distributions of noise standard deviation and model discrepancy hyperparameters based on 3D deformation data of blade ID 01, using offline Bayesian calibration	101
6.9	Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the elongation at the blade tip data of blade ID 90, using offline Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy	102
6.10	Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the 3D deformation data of blade ID 01, using offline Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy	102
6.11	Posterior distributions of model parameters using elongation data at the blade tip (blade ID 90), using online Bayesian calibration	103
6.12	Posterior distributions of noise standard deviation and model discrepancy hyperparameters using elongation data at the blade tip (blade ID 90), using online Bayesian calibration	103
6.13	Posterior distributions of model parameters using 3D deformation data of blade ID 01, using online Bayesian calibration	104
6.14	Posterior distributions of noise standard deviation and model discrepancy hyperparameters based on 3D deformation data of blade ID 01, using online Bayesian calibration	104
6.15	Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the elongation at the blade tip data of blade ID 90, using online Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy	105
6.16	Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the 3D deformation data of blade ID 01, using online Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy	105

7.1	The bond formation process between two filaments: (1) initial surface contact; (2) wetting or neck growth; (3) molecular diffusion and randomization across the cross-section of two FFF extruded filaments	110
7.2	Flowchart of the simulation models for (a) Case A and (b) Case B	114
7.3	Deposition sequence of unidirectional 90 filaments: (a) from left to right for all the layers and (b) from left to right in odd numbered layers and from right to left in even numbered layers	116
7.4	The experimental setup	117
7.5	Top view temperature profile of the first layer	119
7.6	Temperature evolution of 90 filaments at $z_{\text{cut}} = L/2$ (a) with $L = 0.035$ m shown in Fig. 7.3a and (b) with $L = 0.21$ m shown in Fig. 7.3b along deposition time	119
7.7	Experimental temperature profiles compared with model predictions for the interface between filaments 1 and 2 at $z_{\text{cut}} = 0.0175$ m and the neck growth predictions using the heat transfer model predictions (case A) and experimental temperature data (case B)	120
7.8	Sensitivity indices for the bond length between the 1st and 2nd filaments in the first layer	121
7.9	Sensitivity indices for the bond length between the 20th and 21st filaments in the second layer	122
7.10	Sensitivity indices for the bond length between the 50th and 51st filaments in the fourth layer	123
7.11	Sensitivity indices for the bond length between the 87th and 88th filaments in the sixth layer	123
7.12	Prior and posterior distributions of the material property α and model parameter β considering the heat transfer model predictions as the input to the sintering neck growth model (case A)	124
7.13	Prior and posterior distributions of the model parameter β using experimental temperature profile as the input to the sintering neck growth model (case B)	125
7.14	Cross-section view of a sample produced with $T_n = 240^\circ\text{C}$, and $v_p = 0.042$ m/s	126
7.15	Validation of the bond length prediction model for case A, where x and y-axes represent the bond length predictions and experimental observations respectively at each interface between adjacent filaments	127

7.16	Validation of the bond length prediction model for case B, where x and y-axes represent the bond length predictions and experimental observations respectively at each interface between adjacent filaments	127
7.17	Bonding frontier for Ultimaker 2 Extended +	128
7.18	Validation of the optimization results, where x and y-axes represent the bond length predictions and experimental observations respectively at each interface between adjacent filaments of Case I and II, respectively (three specimens printed for each case)	129
7.19	Cross-section views of the parts at $z_{\text{cut}} = 0.0175$ m built with non-optimal and optimal process parameters: (a) sample 1, (b) sample 2, (c) case I and (d) case II	130
7.20	Flowchart of the proposed methodology	139
7.21	Deposition sequence of unidirectional filaments	140
7.22	Design of experiments for process parameters	140
7.23	Thickness measurement of the printed part	141
7.24	Cross-section view of a sample with poor bonding	142
7.25	Part thickness of a sample along Z-axis locations	142
7.26	Predicted bond lengths and uncertainty bounds at randomly selected interfaces from the test data and actual bond length measurements (obtained using microscopy images)	144
7.27	Average bond length predictions vs. observations of three sets of parts printed with same process parameters	144
7.28	Comparison of mean dropout prediction and observation of bond lengths	145
7.29	Comparison of mean dropout prediction and observation of part thickness	145
7.30	Hypervolume values (mean and one standard deviation) of nine different experimental designs of Case 3	147
7.31	Hypervolume values vs. number of generations of Case 3 (Population size 200)	148
7.32	Pareto front of Case 1: $A_1 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (217.09, 26.14, 0.42)$, $B_1 = (244.54, 29.59, 0.60)$, $C_1 = (219.03, 43.96, 0.42)$	148
7.33	Pareto front of Case 2: $A_2 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (272.17, 33.68, 0.42)$, $B_2 = (223.92, 31.02, 0.42)$, $C_2 = (251.41, 36.63, 0.42)$	149
7.34	Pareto front of Case 3: $A_3 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (217.02, 26.01, 0.42)$, $B_3 = (217.05, 27.84, 0.42)$, $C_3 = (274.04, 34.29, 0.42)$	150

7.35	Pareto designs for Case 4: $A_4 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (217.08, 26.88, 0.42)$, $B_4 = (277.99, 39.41, 0.42)$, $C_4 = (225.34, 31.84, 0.42)$	151
7.36	Parallel coordinate representation of the model dependencies with a color mapping along the first objective $(1 - \mu_{\hat{b}_1})$	152
7.37	Parallel coordinate representation of the model dependencies with a color mapping along the third objective $(\mu_{\hat{h}})$	152
7.38	The correlation coefficient between design variables and objective functions	153
7.39	Pareto front and MCS results of Case 1	154
7.40	Pareto front and MCS results of Case 2	154
7.41	Pareto front and MCS results of Case 3	155
7.42	Experimental validation of optimal process design (Case 3)	156

CHAPTER 1

Introduction

1.1 Overview

In analyzing a dynamic multi-physics system, both the computational effort and the accuracy of the system model in predicting the system response present challenges. The challenges become even greater in the case of nonlinear behavior, in systems as different as gas turbine engines and additive manufacturing processes, where the output (system response or product properties, respectively) is governed by a coupled multidisciplinary system of equations (fluid mechanics, heat transfer, and structural mechanics), and whose analysis requires great computational effort. Directly measuring the true response of a complex engineering system for many input realizations by conducting experiments is not affordable, thus computational models are used to analyze the system of interest for a variety of input realizations. However, the computational model is often an incomplete representation of the complex physical system, thus the system response prediction is affected by multiple heterogeneous sources of uncertainty. Therefore it is important to quantify the uncertainty in the predictions from computational models.

Uncertainty quantification (UQ) can be categorized into two problems; the forward problem and the inverse problem. In the forward problem, the uncertainty regarding the model inputs, model parameters, and model errors is propagated and aggregated to compute the uncertainty in the output. Whereas, in the inverse problem, model calibration is performed using the available set of measurements. Both of these problems typically require a large number of model runs. Furthermore, in a complex system the quantity of interest (QoI) is often a multivariate output that is a field quantity (exhibits spatial dependence) and/or a stochastic process (exhibits temporal dependence), which makes it challenging to map the set of model inputs to the output. Therefore, an inexpensive surrogate of the original physics model becomes necessary to perform the forward and inverse analyses. The inexpensive surrogate model also helps to carry out other analyses such as probabilistic diagnosis and prognosis and risk analysis to support the decision making.

The surrogate model can be developed through machine learning (ML) techniques, using data generated from three sources: (1) physics model runs, (2) experiments, or (3) combination of both physics model runs and experiments. In this work, surrogate model construction using all three options will be explored. The surrogate models are physics-informed in different ways for predicting

the QoI while ensuring consistency with physical laws. Three different strategies are explored for incorporating physics knowledge into ML models: (1) incorporate physics constraints within the loss function used in training the ML models, (2) use physics model outputs as additional inputs to the ML model trained with experimental data, and (3) pre-train an ML model with physics model input-output and then update it with experimental data. The first step of pre-training using data generated by the physics model can be thought of as similar to a lower fidelity model, and the second step of improving the pre-trained model (either by parameter updating or by adding a discrepancy term) can be thought of as similar to incorporating higher fidelity data (experimental data, in this case) to improve the model. Thus the third strategy can also be used for constructing surrogate models using physics models of different fidelity.

In this study, we expand the surrogate modeling methodology to multi-physics dynamic problems with high-dimensional spatio-temporal output, where the output QoI is also time-dependent. For high-dimensional applications, the surrogate modeling techniques in the literature have investigated efficiency improvement by reducing the dimensionality in both the input and output spaces. Dimension reduction in the output space has been typically performed using the principal directions in the output data matrix (e.g., principal components analysis (PCA), or singular value decomposition (SVD)) [1–6]. To address the challenge introduced by the high-dimensionality of spatially and temporally varying outputs, a data compression method is first employed to convert the high-dimensional output into low-dimensional latent space. ML models are then constructed for each key feature in the latent space. For input dimension reduction, several methods have been proposed, such as the proper orthogonal decomposition (POD) [7] and projecting inputs to an orthogonal basis like the truncated Karhunen-Loeve expansion [8], or a combination of both strategies (e.g., sparse pseudospectral approximation) [9, 10]. Sparse grids [11] and active subspace transformation of the dataset’s gradient space [12, 13] are some of the other existing methods that are suitable for input dimension reduction in the case of dependent variables, where there is correlation to exploit. Recent efforts have focused on gradient-based techniques to increase the capability of surrogate models to very large input and outputs dimensions involving multivariate outputs, such as the PCAS method [14, 15] that extend the applicability of active subspace methods.

Often the available resources are limited to obtain an adequate amount of training data. Therefore, it is important to construct an accurate surrogate model with the fewest possible experiments or computer simulations. A design of experiments (DoE) [16] approach, which refers to the selection of the inputs at which to conduct these experiments, becomes crucial for sufficiently accurate model construction with minimum computational or experimental cost. One of the most popular DoE

approaches is space-filling methods [17], that covers the input space uniformly without considering the physics information. An alternative DoE approach is the adaptive DoE, which uses the physics information represented by the surrogate model, to sequentially add new training points that are sampled in regions of high interest. This work focuses on adaptive selection of input samples where an initial experimental design is extended sequentially by considering both the physics of the problem of interest and the exploration of the entire design space to improve the model performance within the available resources.

Once an adequately satisfactory surrogate model is constructed, this work develops an approach that fuses multi-source information for Bayesian calibration of multi-physics models with multi-level data that contain both local and shared parameters. The direct use of all the data simultaneously for Bayesian calibration is not practical, especially when there exists multiple types of measurement data from multiple components at different levels [18, 19]. A segmented approach is pursued in this work to integrate expert knowledge and measurements from multiple domains at different levels of the network to support model calibration.

The effects of different sources uncertainty propagate through the physics-based or ML models to the optimization framework, which causes uncertainty in the output QoI. These heterogeneous sources of uncertainty need to be accounted for while optimizing the process design. It is desirable to reduce the epistemic uncertainty by using available information such as experimental data, which reduces the uncertainty propagated to the prediction. Model calibration is performed to reduce the epistemic uncertainty with available measured data. In order to reduce the dimension of model calibration, non-important uncertainty sources that are identified using sensitivity analysis can be fixed. Global sensitivity analysis (GSA) aims to quantify the effects of input random variables (or combinations thereof) on the variance of the response of a physical or mathematical model [20–23]. The sensitivity analysis needs to consider both aleatory and epistemic uncertainty sources. However, sensitivity analysis that include the contribution of both aleatory and epistemic uncertainty such as model error is not well-established [24, 25]. The predictions improved beyond that of physics models or ML models alone and the variability of prediction is reduced by using physics-informed machine learning (PIML) models, thus the confidence in the sensitivity analysis results is increased.

Ultimate, the purpose of uncertainty quantification in the system response prediction is to support decision-making under uncertainty. For example, in additive manufacturing (AM), the necessity to design the manufacturing process parameters with a rigorous understanding of where variability comes from and how to reduce the variability has gained considerable attention. The traditional way is to use a physical experiment-based trial-and-error approach for establishing a relationship between

the process parameters and the QoI. In a trial-and-error approach, the physical process is repeated multiple times with different process parameter combinations to achieve the desired QoI; this is expensive and time-consuming. Moreover, this trial-and-error approach needs to be implemented every time a new design is needed. Therefore, in recent years, research efforts have focused on model-based methods for optimizing the process parameters. The use of physics-based or ML models instead of physical experiments would be more economical in exploring a wide range of process parameter settings and their effects on the product quality. Achieving the desired material properties and product quality in AM processes has been studied using trial-and-error experiments as well as process models (either physics-based or ML models) [26–33]. In this research, both physics-based and ML models are used for AM process parameter optimization under uncertainty, considering both single- and multi-objective formulations.

1.2 Research Objectives

The research efforts in this dissertation are divided into four objectives:

1. Machine learning using both physics knowledge and experimental data
2. Adaptive surrogate modeling for high-dimensional spatio-temporal output
3. Multi-level information fusion for model calibration
4. Decision-making under uncertainty

The overall goal of this research is to develop comprehensive machine learning and optimization models under uncertainty for reducing computational effort while maintaining accuracy. To do this, we first developed several physics-informed and hybrid machine learning models for bond quality and porosity predictions of fused filament fabrication (FFF) parts using physics constraints, physics-based models, and experimental data. The developed physics-informed machine learning (PIML) strategies and their combinations are then investigated for global sensitivity analysis (GSA) using both physics knowledge and experimental data. Physics knowledge and experimental observations are fused in order to maximize the accuracy of sensitivity estimates. Afterwards, an efficient surrogate modeling technique is developed for high-dimensional time-dependent systems with spatio-temporal output and demonstrated for a gas turbine engine rotor. Since 3D finite element models (FEM) for thermo-mechanical systems are computationally expensive, we develop an adaptive sampling strategy that is aimed at reducing the number of computer simulations, thus reducing the cost and time. Next, a Bayesian methodology is developed to fuse information from heterogeneous sources

and account for uncertainties in modeling and measurements for time-dependent multi-component systems. The last objective is developed in the context of additive manufacturing, where the process model error is quantified and included in the process parameter optimization.

1.3 Organization of the Dissertation

The subsequent chapters of this thesis will be devoted to the objectives mentioned above.

Chapter 2 provides background information on the physics-based models for simulating the Fused Filament Fabrication (FFF) process and several state-of-the-art ML algorithms, including: (1) Gaussian process (GP) surrogate modeling, (2) deep neural network (DNN), and (3) Bayesian neural network (BNN). BNN is introduced as a generic means to characterize model prediction uncertainty. Next, variance-based global sensitivity analysis (GSA) is introduced to assess the contribution of each random variable to system-level variability.

Chapter 3 develops several physics-informed machine learning (PIML) models for bond quality and porosity predictions of fused filament fabrication (FFF) parts using physics constraints, physics-based models, and experimental data, thus helping the ML model to produce more accurate and physically meaningful results. Three types of strategies are explored to incorporate physics constraints and multi-physics FFF simulation results into a deep neural network (DNN), thus ensuring consistency with physical laws: (1) incorporate physics constraints within the loss function of the DNN, (2) use physics model outputs as additional inputs to the DNN model, and (3) pre-train a DNN model with physics model input-output and then update it with experimental data. These strategies help to enforce a physically consistent relationship between bond quality and tensile strength, thus making porosity predictions physically meaningful. Eight different combinations of the above strategies are investigated. The results show how the combination of multiple strategies produces accurate ML models even with limited experimental data.

Chapter 4 proposes an approach for information fusion and machine learning for sensitivity analysis using both physics knowledge and experimental data, while accounting for model uncertainty. Variance-based sensitivity analysis is used to quantify the relative contribution of each uncertainty source to the variability of the output quantity. Two types of ML models were considered, namely, GP and DNN models. Several PIML strategies that were developed in Chapter 3 are used to estimate the Sobol' indices. The results show that the application of PIML strategies to both GP and DNN enables accurate Sobol' index computations even with smaller amounts of experimental data while producing physically meaningful results.

Chapter 5 investigates methods for adaptive surrogate model building for multivariate spatio-

temporal output. A cross-validation strategy is used to identify the most accurate surrogate model in the lower dimensional space. Once the most accurate surrogate model type is identified for the problem of interest, the prediction error in the original space is evaluated with different error metrics. A novel adaptive sampling technique, which combines exploration and exploitation for adaptive learning in multi-physics dynamic systems with high-dimensional spatio-temporal outputs, is introduced to improve the surrogate model accuracy with the fewest possible runs of the expensive physics-based model. The effectiveness of the proposed methodology is further demonstrated for a turbine blade example using a time-dependent multi-physics dynamic system model with high-dimensional spatio-temporal outputs.

Chapter 6 formulates a multi-level Bayesian calibration methodology for a multi-component system. Multi-level Bayesian calibration is performed to estimate local and global parameters using experimental data that is obtained at different time instances of different system components. The multi-level Bayesian calibration is pursued in two directions: (1) Offline Bayesian calibration, and (2) Online Bayesian calibration. In the former strategy, the calibration is performed offline using data that is collected over multiple time steps. In the latter strategy, the calibration is performed in real-time as new measurements are obtained at each time step. The updated model is improved with the accumulation of new data. Expert knowledge and experimental data in different levels of the network enable the calibration process in an efficient manner even with insufficient data. Analysis models and observation data for the thermo-mechanical behavior of gas turbine engine blades are used to analyze the effectiveness of the proposed approach.

Chapter 7 presents an optimization framework for decision-making under uncertainty, and demonstrates it for process parameter optimization in additive manufacturing, specifically fused filament fabrication (FFF). The multi-objective optimization under uncertainty is pursued in two directions: (1) Single quantity of interest (QoI), and (2) Two quantities of interest (QoIs). The proposed framework is composed of four elements: (1) Experiments are conducted in the laboratory to print parts with varying process parameters, and data is collected to measure quality characteristics of the parts. (2) Physics-based or data-driven ML models are constructed. (3) The models are then used to find the optimal process parameters, and several multi-objective problem formulations are investigated for robust design optimization (RDO). Uncertainty in the prediction model (including model error) and input variability are considered in the optimization. (4) Finally, Pareto surfaces are constructed to estimate the trade-offs between the objectives. The effectiveness of the proposed methodology is validated by manufacturing the parts at optimal settings and demonstrating the quality improvement.

The conclusions and suggestions for future research are presented in Chapter 8.

CHAPTER 2

Background

This chapter first introduces dimension reduction techniques, then describes several machine learning techniques that are extensively used in the literature. Next, we briefly review uncertainty quantification, global sensitivity analysis and fused filament fabrication (FFF). Following this structure, two dimension reduction techniques are defined in Section 2.1. Afterwards, the basic concept of Gaussian process (GP) surrogate modeling is introduced in Section 2.2.1. In parallel, two deep learning algorithms: deep forward neural network and Bayesian neural network, as a generic means to characterize the prediction uncertainty of neural networks, are introduced in Sections 2.2.2 and 2.2.3. In Section 2.3, we briefly review Bayesian calibration, particle filter and Bayesian network. In Section 2.4, we describe global sensitivity analysis to measure the contribution of a random variable to the system-level variability. Finally, Section 2.5 defines the fused filament fabrication (FFF) additive manufacturing (AM) process and the coupled multi-physics models used to describe the FFF process.

2.1 Dimension Reduction

2.1.1 Singular Value Decomposition (SVD)

Singular value decomposition (SVD) is a generalized eigen-decomposition technique and multivariate statistical method for describing a large amount of high-dimensional data by mapping to a low-dimensional latent space [34]. SVD is applicable to non-square matrices and can be used to handle the spatial correlation of the response. Given n_s data points (i.e., total number of nodes) over the spatial domain Ω for n_t time domain realizations and for n_d realizations of design domain (i.e., training points), a data matrix for the k -th output can be written as

$$\begin{aligned} \mathbf{D}_k &= [\mathbf{D}_k(t_i, \theta_1), \mathbf{D}_k(t_i, \theta_2), \dots, \mathbf{D}_k(t_i, \theta_{n_d})]^T \\ &= \begin{bmatrix} D_k(t_i, s_1, \theta_1) & D_k(t_i, s_1, \theta_2) & \dots & D_k(t_i, s_1, \theta_{n_d}) \\ D_k(t_i, s_2, \theta_1) & D_k(t_i, s_2, \theta_2) & \dots & D_k(t_i, s_2, \theta_{n_d}) \\ \vdots & \vdots & \ddots & \vdots \\ D_k(t_i, s_{n_s}, \theta_1) & D_k(t_i, s_{n_s}, \theta_2) & \dots & D_k(t_i, s_{n_s}, \theta_{n_d}) \end{bmatrix}^T \end{aligned} \quad (2.1)$$

where $\mathbf{D}_k(t_i, \theta_j) = [D_k(t_i, s_1, \theta_j), D_k(t_i, s_2, \theta_j), \dots, D_k(t_i, s_{n_s}, \theta_j)]$ is the i -th temporal location of the j -th realization for the k -th field response, t_i represents the i -th temporal location and θ stands for different realizations in the design domain.

The response \mathbf{S} of the original physics-based model can be collected at the training points as

follows:

$$\mathbf{S} = [\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_{n_k}]^T \quad (2.2)$$

where n_k is the total number of physics model outputs.

This large amount of high-dimensional response can be mapped to a low-rank approximation by using SVD as $\mathbf{S} = \mathbf{U}\mathbf{M}\mathbf{V}^T$, where \mathbf{U} is a $(n_t \times n_d) \times n_s$ matrix, \mathbf{V} is a $n_s \times n_s$ orthogonal matrix and \mathbf{M} is a $n_s \times n_s$ rectangular diagonal matrix with non-negative real numbers $\lambda = [\lambda_1, \lambda_2, \dots, \lambda_r]$, $r = \min((n_t \times n_d), n_s)$ in the diagonal. The diagonal elements λ of \mathbf{M} are called singular values and are arranged in descending order. The number of important features $q (q \leq r)$ to be used to reconstruct the data matrix are determined based on the magnitudes of the singular values. The original data matrix \mathbf{S} can be reconstructed as $\tilde{\mathbf{S}}$ by using the first q largest singular values

$$\mathbf{S}(t_i, \theta_j)^T \approx \tilde{\mathbf{S}}(t_i, \theta_j) = \sum_{p=1}^q \xi_p(t_i, \theta_j) \mathbf{V}_p, \quad (2.3)$$

$$\forall i = 1, 2, \dots, n_t; j = 1, 2, \dots, n_s,$$

where $\mathbf{S}(t_i, \theta_j)^T$ is the $(i \times j)$ -th row of \mathbf{S} , $\xi_p(t_i, \theta_j)$ is the element of $\xi = \mathbf{U}\mathbf{M}$ at $(i \times j)$ -th row and p -th column, and \mathbf{V}_p is the p -th important feature vector used to approximate \mathbf{S} .

It is not only computationally expensive but also memory intensive to directly perform deterministic SVD on a large matrix. Thus, the randomized SVD (rSVD) algorithm, which avoids the high computational cost while not sacrificing accuracy, is used to obtain a low-rank approximation of this large response matrix [35].

2.1.2 Randomized SVD (rSVD)

The randomized SVD (rSVD) method [35] maps the original data matrix onto a small random subspace. Thus, the most important characteristics of the original matrix \mathbf{S} are condensed into a small randomized subspace to obtain an approximate matrix decomposition while avoiding the high computational cost inherent in deterministic SVD.

The main computational operations in rSVD are matrix-matrix multiplications, QR decomposition and SVD on small matrices. More specifically, the rSVD involves five main components: (1) constructing a random matrix; (2) projection of the original data matrix with the random matrix; (3) QR decomposition of the resulting matrix; (4) multiplying the original data matrix with the resulting orthogonal basis; and (5) deterministic SVD on resulting matrix.

In order to obtain a low-rank, say rank r ($r \ll N$), approximation of a matrix $\mathbf{X} \in \mathbb{F}^{M \times N}$,

$$\mathbf{X} \approx \mathbf{U}_r \Sigma_r \mathbf{V}_r' \quad (2.4)$$

the target dimension, i.e., the first r pairs of singular value and singular vectors, can be obtained with the following steps. First, a random projection matrix $\mathbf{P} \in \mathbb{R}^{N \times r}$ is used to get an orthonormal basis for the data matrix \mathbf{X} ,

$$\mathbf{Z} = \mathbf{X}\mathbf{P}, \quad (2.5)$$

where $\mathbf{Z} \in \mathbb{R}^{M \times r}$ is a much smaller matrix ($r \ll N$) than \mathbf{X} and approximate the column space of \mathbf{X} with high probability due to the randomness of \mathbf{P} . Then, a QR decomposition, which is normally employed to compute the SVD, of \mathbf{Z} can be obtained:

$$\mathbf{Z} = \mathbf{Q}\mathbf{R}^{QR}, \quad (2.6)$$

where $\mathbf{Q} \in \mathbb{R}^{M \times r}$ and $\mathbf{R}^{QR} \in \mathbb{R}^{r \times r}$. Then, the data matrix \mathbf{X} is projected onto the subspace \mathbf{Q} and SVD is performed on the projection $\mathbf{Y} \in \mathbb{R}^{r \times N}$ to obtain Σ_r and \mathbf{V}_r^T :

$$\begin{aligned} \mathbf{Y} &= \mathbf{Q}^T \mathbf{X}, \\ \mathbf{Y} &= \mathbf{U}_Y \Sigma_r \mathbf{V}_r^T. \end{aligned} \quad (2.7)$$

The matrices Σ_r and \mathbf{V}_r^T in Eq. (2.7) are the same for \mathbf{X} since \mathbf{Q} approximates the column space of \mathbf{X} [35]. Thus, the r left singular vectors $U_r \in \mathbb{R}^{M \times r}$ of \mathbf{X} is

$$\mathbf{U}_r = \mathbf{Q}\mathbf{U}_Y. \quad (2.8)$$

A lower dimensional representation (dimension r) in place of the original data (dimension N) can be taken as

$$\hat{\mathbf{X}}^{LD} = \mathbf{U}_r \Sigma_r, \quad (2.9)$$

Randomized SVD is also a linear mapping like the basic SVD. Since \mathbf{Z} is of a much smaller size compared to \mathbf{X} and the basis \mathbf{Q} is low rank, the QR decomposition in Step (3) and the deterministic SVD in Step (5) are not time consuming. This makes randomized SVD a powerful tool when dealing with larger data matrix. The time complexity of rSVD is $O(MN \log(r) + (M+N)r^2)$.

2.2 Machine Learning (ML)

2.2.1 Gaussian Process (GP) Surrogate Modeling

A Gaussian process (GP) surrogate model (or Kriging) [36] approximates a response function $y = G(\mathbf{x})$ over the domain of input \mathbf{x} as a Gaussian random process with a mean function $m(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$, which describes the deviation of the model from the trend

$$\mathbf{y}(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')). \quad (2.10)$$

Given a set of training data $\{\mathbf{X}_T, \mathbf{Y}_T\}$ and the input \mathbf{X}_P , where prediction is desired, the conditional probability distribution of the output \mathbf{Y}_P follows a multivariate Gaussian distribution [36] as

$$\begin{aligned} \mathbf{Y}_P | \mathbf{X}_P, \mathbf{X}_T, \mathbf{Y}_T &\sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ \boldsymbol{\mu} &= m(\mathbf{X}_P) + \boldsymbol{\Sigma}_{PT}(\boldsymbol{\Sigma}_{TT} + \sigma_{obs}^2 \mathbf{I})^{-1}(\mathbf{Y}_T - m(\mathbf{X}_T)) \\ \boldsymbol{\Sigma} &= \boldsymbol{\Sigma}_{PP} - \boldsymbol{\Sigma}_{PT}(\boldsymbol{\Sigma}_{TT} + \sigma_{obs}^2 \mathbf{I})^{-1}\boldsymbol{\Sigma}_{PT}^T \end{aligned} \quad (2.11)$$

where $\boldsymbol{\mu}$ is the mean vector of the prediction \mathbf{Y}_P conditioned on the training data, and $\boldsymbol{\Sigma}$ is the conditional covariance matrix of \mathbf{Y}_P ; $\boldsymbol{\Sigma}_{PT}$ is the covariance matrix between the prediction data \mathbf{X}_P and each of the training points $\{\mathbf{X}_T = \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$; $\boldsymbol{\Sigma}_{TT}$ is the $n \times n$ covariance matrix of the training data; $\boldsymbol{\Sigma}_{PP}$ is the unconditional covariance matrix of \mathbf{Y}_P ; σ_{obs}^2 is the variance of observation/measurement error (also called noise variance), and \mathbf{I} is the identity matrix.

The mean function $m(\cdot)$ could be constant, linear or a non-linear function depending on the problem [26, 36–38]. The covariance function $k(\cdot)$ can be formulated by using a covariance function based on the desired properties (order of continuity, stationary/non-stationary, isotropic/anisotropic). A squared exponential correlation function with separate length scale parameters l_i for each input dimension has often been used in the literature:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp \left[-\sum_{i=1}^M \frac{(x_i - x'_i)^2}{2l_i} \right] + \sigma_{obs}^2 \boldsymbol{\delta}(\mathbf{x}, \mathbf{x}') \quad (2.12)$$

where σ_f^2 is the signal/process variance and defines the maximum allowable covariance, and $\boldsymbol{\delta}(\mathbf{x}, \mathbf{x}')$ is the Kronecker delta function. Other correlation functions are also available in the literature, such as linear, exponential, Matérn functions [36, 37, 39].

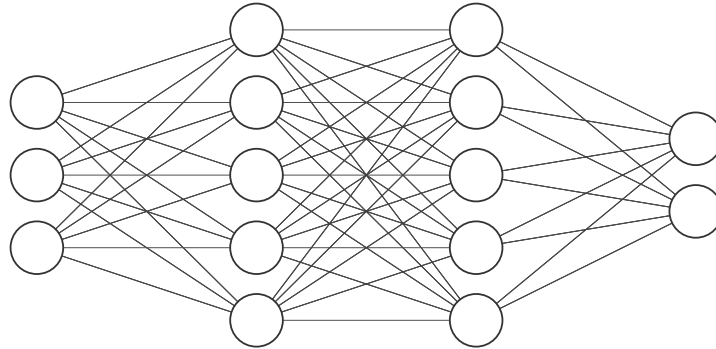
The hyperparameters of the GP model considering a zero mean function, i.e., $\Lambda = \{l, \sigma_f, \sigma_{obs}\}$,

are inferred from the training data. A common method is to maximize the log marginal likelihood function, which is defined as

$$\log p(\mathbf{Y}_T|\mathbf{X}_T;\Lambda) = -\frac{1}{2}\mathbf{Y}_T(\Sigma_{TT} + \sigma_{obs}^2I)^{-1}\mathbf{Y}_T - \frac{1}{2}\log|\Sigma_{TT} + \sigma_{obs}^2I| - \frac{n}{2}\log 2\pi. \quad (2.13)$$

2.2.2 Deep Neural Network (DNN)

In recent years, due to the confluence of advanced sensing and imaging techniques, big data processing techniques, enormous computational power and the internet, rapid advances are being made in developing sophisticated data-driven machine learning models, particularly neural networks. A deep neural network (DNN) is composed of multiple hidden layers and has four major components: neuron, activation function, cost function, and optimization. Figure 2.1 shows a neural network consisting of three inputs, two hidden layers, each having four neurons, and two output neurons. The values of various input variables of a particular neuron are multiplied by their associated weights, then the sum of the products of the neuron weights and the inputs are calculated at each neuron. The summed value is passed through an activation function that maps the summed value to a fixed range before passing these signals on to the next layer of neurons.



Input Layer $\in \mathbb{R}^3$ Hidden Layer $\in \mathbb{R}^5$ Hidden Layer $\in \mathbb{R}^5$ Output Layer $\in \mathbb{R}^2$

Figure 2.1: A fully connected deep neural network (DNN) with two hidden layers

The predictions of the DNN after forward propagation, $\hat{\mathbf{Y}}$, are compared against the observations, \mathbf{Y}_{obs} , by defining a loss function (e.g., root mean squared error (RMSE); $\mathcal{L}_{RMSE}(\mathbf{Y}_{obs}, \hat{\mathbf{Y}}) = \sqrt{\sum_{i=1}^n (y_{obs,i} - \hat{y}_i)^2/n}$), which measures how far off the predictions are from the observations for the n training samples. Backpropagation algorithms are employed to keep track of small perturbations to the weights that affect the error in the output and to distribute this error back through the network layers by computing gradients for each layer using the chain rule. In order to minimize the value of the loss function, necessary adjustments are applied at each iteration to the neuron weights in

each layer of the network. These procedures are performed at each iteration until the loss function converges to a stable value.

2.2.3 Bayesian Neural Network (BNN)

In the Bayesian context, the distributions of the model parameters (neuron weights \mathbf{w}) that are most likely to generate the observed data are inferred. A prior distribution over the neuron weights $p(\mathbf{w})$ is defined, and a likelihood function $p(\mathbf{Y}|\mathbf{X}, \mathbf{w})$ is defined to represent the probability of generating the observed data given model parameters. Bayesian neural network (BNN) implementations so far have used Gaussian prior distributions $p(\mathbf{w} = \mathcal{N}(0, \mathcal{I}))$ to replace the deterministic network’s weight parameters [40–42], thus representing the epistemic uncertainty in the DNN model. Following Bayes’ theorem, a posterior distribution over the model parameters given the training set $\{\mathbf{X}, \mathbf{Y}\} = \{\{\mathbf{x}_1, \dots, \mathbf{x}_N\}, \{\mathbf{y}_1, \dots, \mathbf{y}_N\}\}$ is defined by

$$p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{Y}|\mathbf{X})}. \quad (2.14)$$

After inferring the posterior distributions of $p(\mathbf{w})$, the predictive distribution of the model output for a new input \mathbf{x}^* is

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int_{\Omega} p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})p(\mathbf{w}|\mathbf{X}, \mathbf{Y})d\mathbf{w}. \quad (2.15)$$

The posterior distribution of model parameters $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ cannot be evaluated analytically over the whole parameter space Ω due to the highly non-linear behavior in the neural network caused by the non-linear activation functions and their combinations across multiple hidden layers. Thus, it becomes difficult to perform exact analytical inference in BNNs.

Several approximate inference techniques are available to infer the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$, such as variational inference [43], Approximate Bayesian Computation [44], Markov Chain Monte Carlo (MCMC) [45], and Particle Filter [46] methods. Markov Chain Monte Carlo (MCMC) can be a more accurate technique than the other methods, but it is computationally too expensive. It would not be affordable to perform MCMC to estimate posterior distributions of the deep neural network model parameters (i.e., weights) since there are usually a large number of parameters in the model. Variational inference (VI) fits a simple and tractable distribution $q_{\theta}(\mathbf{w})$ to the posterior, parametrized by a variational parameter θ [43]. This approximates the intractable problem by optimizing the parameters of $q_{\theta}(\mathbf{w})$. The accuracy of the variational distribution is often measured by the Kullback-Leibler (KL) divergence between the approximate distribution $q_{\theta}(\mathbf{w})$ and the true model posterior $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$.

Gal and Ghahramani [47] showed that Monte Carlo (MC) dropout is equivalent to performing approximate VI; the former infers the posterior by performing dropout not only while training a model but also during prediction. In MC dropout, randomly chosen neurons are temporarily removed from the network along with their connections. Next, the gradients of neuron weights are calculated on each smaller neural network and these gradients are then averaged over the training sets to obtain the weights of overall network. The construction of BNNs with MC dropout builds on the concept of dropout as regularization on neural networks. However, in contrast to standard neural networks, MC dropout performs dropout and generates random samples following a Bernoulli distribution for each neuron in the input and hidden layers during prediction. The dropout is applied to the neuron that takes the value 0 with a given dropout probability p_d . The outputs of the network are predicted using the collection of generated random samples from the posterior predictive distribution and the uncertainty in the prediction is quantified. The computational effort for the construction of BNNs with MC dropout is comparable to the standard neural networks. Moreover, the simplicity of the MC dropout strategy provides an efficient way of Bayesian inference to quantify the model prediction uncertainty in a variety of neural networks, such as DNN, convolutional neural network (CNN), and recurrent neural network (RNN).

2.3 Uncertainty Quantification

The output of any model is affected by various sources of uncertainty. The uncertainty sources can be aleatory (natural variability) or epistemic (lack of knowledge). Epistemic uncertainty is caused by insufficient knowledge or information about the model and data. If limited data is available, there is epistemic uncertainty in the model prediction. Aleatory uncertainty is caused by the natural variability in the process, leading to variability in the process output QoI. When multiple models are used to predict the system behavior, the uncertainty propagates from each model to another as a function of model coupling. Uncertainty quantification (UQ) is an important step toward quantifying and reducing these heterogeneous sources of uncertainty. UQ is a process of quantifying and investigating the effects of these different uncertainty sources on the QoIs.

UQ can be performed in two ways: the forward problem and the inverse problem. In the forward problem, model errors and the uncertainty related to the model inputs and parameters are propagated to compute the uncertainty in the output. On the other hand, model calibration is performed using the available measurement data in the inverse problem. Several model calibration techniques are available in the literature (e.g., least squares, maximum likelihood estimation, and Bayesian estimation). Bayesian methods provide a convenient framework for combining prior beliefs about

parameters with current evidence gained from data [48, 49]. In the inverse problem, unmeasured inputs or model parameters can be estimated using Bayesian inference, given observations of the outputs. As a result, they help the inference of unknown parameters by using the observed data on quantities affected by the unknown parameters. The inverse problem is an essential part of uncertainty quantification and reduction and in achieving the desired level of prediction confidence.

Consider a physics model $G(\cdot)$ that maps input variables \mathbf{X} and model parameters θ_m to the numerical model output \mathbf{Y}_m :

$$\mathbf{Y}_m(\mathbf{X}) = G(\mathbf{X}; \theta_m(\mathbf{X})) \quad (2.16)$$

Let n_D be the number of collected observation data \mathbf{Y}_{obs} from experiments with input variable settings $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n_D)}$, where $\mathbf{x}^{(i)}$ is the input variable setting for the i th experiment. The difference between observations \mathbf{Y}_{obs} and the true response of the system \mathbf{Y}_{true} is attributed to measurement error $\boldsymbol{\varepsilon}_{\text{obs}}$, which is often treated as a zero-mean Gaussian random variable with variance σ_{obs}^2

$$\mathbf{Y}_{\text{obs}}(\mathbf{X}) = \mathbf{Y}_{\text{true}}(\mathbf{X}) + \boldsymbol{\varepsilon}_{\text{obs}}(\mathbf{X}). \quad (2.17)$$

The physics model prediction is inaccurate due to missing physics or due to other approximations. Thus, an additive uncertainty model with a model discrepancy term $\boldsymbol{\delta}(\mathbf{X})$ as a function of model inputs is introduced as shown in Fig. 2.2 to capture the difference between \mathbf{Y}_m and the true response of the system \mathbf{Y}_{true} [50]:

$$\mathbf{Y}_{\text{true}}(\mathbf{X}) = \mathbf{Y}_m(\mathbf{X}) + \boldsymbol{\delta}(\mathbf{X}) \quad (2.18)$$

Combining Eqs. (2.17) and (2.18), the overall prediction \mathbf{Y}_{pred} that accommodates various errors can be written as

$$\begin{aligned} \mathbf{Y}_{\text{true}}(\mathbf{X}) &= \mathbf{Y}_{\text{obs}}(\mathbf{X}) - \boldsymbol{\varepsilon}_{\text{obs}}(\mathbf{X}) = \mathbf{Y}_m(\mathbf{X}) + \boldsymbol{\delta}(\mathbf{X}) \\ \mathbf{Y}_{\text{pred}}(\mathbf{X}) &= \mathbf{Y}_m(\mathbf{X}) + \boldsymbol{\delta}(\mathbf{X}) + \boldsymbol{\varepsilon}_{\text{obs}}(\mathbf{X}) \end{aligned} \quad (2.19)$$

A common approach to estimate the discrepancy term $\boldsymbol{\delta}(\mathbf{X})$ is the one formulated by Kennedy and O’Hagan [50], which is applicable in the context of Bayesian calibration (see Section 2.3.1). The discrepancy term can be expressed in multiple ways, such as constant, Gaussian random variable with unknown parameters (either input-dependent or not), or Gaussian process (either stationary or non-stationary) [51]. The hyperparameters of the discrepancy term are then estimated along with

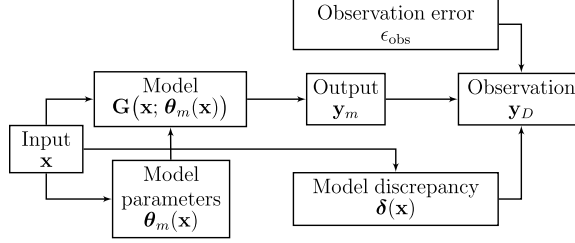


Figure 2.2: Relating model output to observation data

the physics model parameters using Bayesian calibration [26].

2.3.1 Bayesian Calibration

The unknown model parameters $\boldsymbol{\psi}$ and errors can be estimated using the experimental data, which contain measurement error $\boldsymbol{\epsilon}_{obs}$, through Bayesian calibration. The purpose of Bayesian model calibration is to use observation data \mathbf{Y}_{obs} to estimate the posterior distributions of unknown parameters such as model parameters, measurement error and the model discrepancy term $\boldsymbol{\delta}$ (if applicable). The measurement error $\boldsymbol{\epsilon}_{obs}$ is commonly represented as a zero-mean Gaussian random variable with an unknown variance σ_{obs}^2 , i.e., $\boldsymbol{\epsilon}_{obs} \sim N(0, \sigma_{obs}^2)$. The model discrepancy $\boldsymbol{\delta}$ can be formulated in various ways depending on the problem and can be a function of the input or any other parameter [51, 52]. $\boldsymbol{\delta}$ may be modeled as a constant bias term, or a random variable with either fixed or input-dependent mean and variance, or even a random process [26, 38, 53, 54]. When the model parameters $\boldsymbol{\psi}$ are uncertain, the calibration parameters $\boldsymbol{\Theta} = [\boldsymbol{\psi}, \Delta, \sigma_{obs}]$ where Δ denotes the parameters of $\boldsymbol{\delta}$. Based on Bayes' theorem, the joint distribution of the calibration parameters is given by

$$f(\boldsymbol{\Theta}|\mathbf{Y}_{obs}) = \frac{f(\mathbf{Y}_{obs}|\boldsymbol{\Theta})f(\boldsymbol{\Theta})}{\int f(\mathbf{Y}_{obs}|\boldsymbol{\Theta})f(\boldsymbol{\Theta})d\boldsymbol{\Theta}} \quad (2.20)$$

where \mathbf{Y}_{obs} is the observation data, $f(\mathbf{Y}_{obs}|\boldsymbol{\Theta})$, $f(\boldsymbol{\Theta})$, and $f(\boldsymbol{\Theta}|\mathbf{Y}_{obs})$ are the likelihood function, joint probability density function (PDF) of $\boldsymbol{\Theta}$, and the joint posterior PDF.

Bayesian model calibration is often performed using Markov chain Monte Carlo (MCMC) sampling algorithms (such as Metropolis-Hastings [55], Gibbs [56], slice sampling [57]) or Particle Filter (PF) [46] since the integral in the denominator of Eq. (2.20) makes numerical integration intractable for increasing dimension of calibration quantities [58].

2.3.2 Particle Filter

Particle filter (PF) [46] is a generic approximate algorithm to track the evolution of the state variables in a dynamic Bayesian network (DBN). A Bayesian network (BN) is a directed acyclic

graph model that represents the joint distribution of a set of random variables and uncertainty in the field of interest. The BN can be extended to a DBN to track a time-dependent system whose states evolve over time, which can be realized as a series of time-independent static BNs. In the literature, PF is also referred as “survival of the fittest”, in which a particle with higher weight that is defined based on likelihood is likely to be re-sampled. Whereas, a particle with lower weight is likely to be discarded. PF can be applied to both discrete and continuous DBNs.

Assume that the state variables at time t , \mathbf{X}_t evolve from the state variables at previous time $t-1$, \mathbf{X}_{t-1} through a nonlinear state function $f: \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x}$:

$$\mathbf{X}_t = f(\mathbf{X}_{t-1}, v_{t-1}) \quad (2.21)$$

where v_{t-1} is the process noise in the state function, the capital letters denote random variables. The measurement $\mathbf{Z}_t \in \mathbb{R}^{n_z}$ is obtained using the measurement function $h \in \mathbb{R}^{n_z \times n_x}$

$$\mathbf{Z}_t = h(\mathbf{X}_t, n_t) \quad (2.22)$$

where $n_t \in \mathbb{R}^{n_z}$ is the measurement noise.

The most basic PF algorithm to track the evolution of the state variables and measurement is sequential importance sampling [46], in which the joint posterior distribution at time t is defined by $p(\mathbf{X}_{0:t} | \mathbf{Z}_{1:t})$. Weights are associated with each particle (i.e., $\{\mathbf{x}_{0:t}^i, w_t^i\}_{i=1}^N$) to approximate the joint distribution

$$p(\mathbf{X}_{0:t} | \mathbf{Z}_{1:t}) \approx \sum_{i=1}^N w_t^i \delta_{\mathbf{x}_{0:t}^i} \quad (2.23)$$

where $\delta_{\mathbf{x}_{0:t}^i}$ is a delta function at $\mathbf{x}_{0:t}^i$, the lower-case letters denote particles and the superscript i indicates that it is the i th particle.

A proposal density is used to sample the i th particle at time t \mathbf{x}_t^i of the state variable \mathbf{X}_t based on the current state $\mathbf{X}_{0:t-1}^i$ and the measurement $\mathbf{Z}_{1:t}$

$$\mathbf{X}_t^i \sim q(\mathbf{X}_t | \mathbf{X}_{0:t-1}^i, \mathbf{Z}_{1:t}). \quad (2.24)$$

The initial weight for each particle is equal, i.e., $1/N$ and the initial state are sampled from the joint prior distribution. The updated importance weights for each particle is given by

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{Z}_t | \mathbf{X}_t^i) p(\mathbf{X}_t^i | \mathbf{X}_{t-1}^i)}{q(\mathbf{X}_t^i | \mathbf{X}_{t-1}^i, \mathbf{Z}_t)}. \quad (2.25)$$

A common problem of PFs is degeneracy, in which the performance of the filter deteriorates after some time steps and importance weights are unevenly distributed, i.e., only a few particles have significant weights, but most other particles have negligible weights. A measure of degeneracy is the effective sample size [59] $N_{eff} = 1/\sum_i^N (w_t^i)^2$. The effect of degeneracy is explained by N_{eff} . Smaller values of N_{eff} indicate severe degeneracy.

This degeneracy problem can be overcome with the re-sampling procedure by multiplying particles with significant weights while discarding low weighted particles (referred to Sampling-Importance Re-sampling, SIR [46]) to force particles to areas of high likelihood. Re-sampling is performed when N_{eff} falls below a user-defined threshold. The simplest strategy for re-sampling is to generate a new set based on the discrete approximation given in Eq. (2.23), and the weights are reset to $w_t^i = 1/N$ again and thus become uniform. The SIR algorithm 1) takes the state transition distribution $p(\mathbf{X}_t|\mathbf{X}_{t-1}^i)$ as the proposal density distribution $q(\mathbf{X}_t|\mathbf{X}_{0:t-1}^i, \mathbf{Z}_{1:t})$, and 2) conducts re-sampling at each iteration, which reduces Eqs. (2.24) and (2.25) to

$$\mathbf{X}_t^i \sim p(\mathbf{X}_t|\mathbf{X}_{t-1}^i), \quad (2.26)$$

$$w_t^i \propto p(\mathbf{Z}_t|\mathbf{X}_t^i). \quad (2.27)$$

Although the re-sampling strategy can reduce the effects of degeneracy, it may also lead to the problem of sample impoverishment. Sample impoverishment can be viewed as highly concentrated particles, while the sample degeneracy is the result of widely distributed particles. As the particles with higher weights are selected multiple times and the particles with lower weights are discarded, sample diversity is not maintained. This phenomenon, known as sample impoverishment [46] becomes apparent when the system is noise-free or has a very small process noise. Therefore, the process noise in Eq. (2.21) plays a crucial role by giving particles the opportunity to move in space stochastically.

In most applications, the model of interest depends on unknown static parameters that need to be estimated from the data. In the case of parameter estimation, the sample impoverishment can be avoided by perturbing the re-sampled particles to be used at the next time step [60, 61]. SIR particle filtering estimates the posterior distribution of parameters via re-sampling and parameter perturbation to avoid degeneracy and sample impoverishment respectively.

2.3.3 Bayesian Networks

A Bayesian network (BN) is a directed acyclic graph (DAG) representation of a multivariate distribution, consisting of nodes and arcs, where nodes represent the random variables in the system and arcs (directed edges between nodes) are associated with conditional probabilities relating the nodes. In other words, a BN decomposes the joint probability distribution of a set of variables into a set of conditional and marginal probabilities [62]. Given random variables $X = \{X_1, X_2, \dots, X_n\}$, the joint probability of these variables is expressed through a BN as

$$Pr(X) = \prod_{i=1}^n Pr(X_i | \Pi_{X_i}) \quad (2.28)$$

where Π_{X_i} denotes the set of parent nodes of X_i and $Pr(X_i | \Pi_{X_i})$ represents the conditional probability distribution of X_i , given its parent nodes. If X_i has no parent nodes (i.e., $Pr(X_i | \Pi_{X_i}) = Pr(X_i)$), then X_i is a root node and is defined by a marginal distribution.

2.4 Global Sensitivity Analysis (GSA)

Consider a deterministic real integrable one-to-one system response function $Y = f(X)$, where $f(\cdot)$ is the computational model, $X = \{X_1, \dots, X_k\}$ are mutually independent model inputs, and Y is the model output. As shown in [63], the variance of Y can be decomposed as

$$V(Y) = \sum_i^k V_i + \sum_{i_1}^k \sum_{i_2=i_1+1}^k V_{i_1 i_2} + \sum_{i_1}^k \sum_{i_2=i_1+1}^k \sum_{i_3=i_2+1}^k V_{i_1 i_2 i_3} + \dots + V_{12\dots k} \quad (2.29)$$

where V_i is the variance of Y due to X_i alone, and $V_{i_1 \dots i_p}$ ($p \geq 2$) indicates the variance of Y caused by the interaction of $\{X_{i_1}, \dots, X_{i_p}\}$.

The Sobol' indices are defined by dividing both sides of Eq. (2.29) with $V(Y)$

$$1 = \sum_i^k S_i + \sum_{i_1}^k \sum_{i_2=i_1+1}^k S_{i_1 i_2} + \sum_{i_1}^k \sum_{i_2=i_1+1}^k \sum_{i_3=i_2+1}^k S_{i_1 i_2 i_3} + \dots + S_{12\dots k} \quad (2.30)$$

where S_i is the first-order or main effects index that assesses the contribution of X_i individually to the variance of the output Y without considering interactions with other inputs. The higher-order indices $S_{i_1 \dots i_p}$ ($p \geq 2$) in Eq. (2.30) measure the contributions of the interactions of $\{X_{i_1}, \dots, X_{i_p}\}$.

The first-order index S_i is defined as follows:

$$S_i = \frac{V_i}{V(Y)} = \frac{V_{X_i}(E_{X_{-i}}(Y|X_i))}{V(Y)} \quad (2.31)$$

where X_{-i} are all the model inputs other than X_i .

The overall contribution of X_i considering an individual input and its interactions with all other inputs is measured by the total effects index S_i^T :

$$S_i^T = 1 - \frac{V_{-i}}{V(\mathbf{Y})} = \frac{V_{X_{-i}}(E_{X_i}(\mathbf{Y}|X_{-i}))}{V(\mathbf{Y})}. \quad (2.32)$$

The computation of S_i analytically is nontrivial since $E_{X_{-i}}(\cdot)$ requires multidimensional integrals. A straightforward model-based approach to estimate Sobol' indices is to use a double-loop Monte Carlo simulation (MCS) [63]. In order to reduce the cost associated with the double-loop MCS, analytical, spectral and efficient sampling-based methods have been developed. The methodology developed by Sudret [64] approximates the original physics model by a polynomial chaos expansion (PCE) model and estimates the Sobol' indices by using the PCE coefficients. Chen et al. [65] proposed analytical formulas to compute Sobol' indices using a GP surrogate model with input variables that follow normal or uniform distributions. The improved FAST method [66] combines the classical FAST method [21] with random balanced design [67] for generating samples to evaluate Sobol' indices. One of these approaches of particular relevance to this dissertation is to replace the original computational model by a surrogate model and use this surrogate model in GSA [68–72].

2.5 Fused Filament Fabrication (FFF)

This section describes the two coupled multi-physics models (thermal model and polymer sintering model) that predict the bond formation between adjacent filaments and the mesostructure of the printed part. The porosity and bond quality of an FFF part is dependent on the temperature history of filaments. Thus, it is important to predict the temperature evolution of filaments to estimate the final mesostructure of the printed part. The thermal model, based on the work by Costa et al. [73], is used to predict the temperature evolution of filaments considering the material properties, part geometry, and process parameters. The output of the heat transfer model (temperature) is input to the sintering model to predict the porosity and bond quality. A new method is then developed, which considers realistic filament geometry, and allows the filament geometry to change during the printing process, to compute the rate of polymer sintering and the final mesostructure of the printed part using the predicted temperature evolution of each filaments. Thus the mapping from input to output is a multi-physics model, i.e., models of two physical phenomena (heat transfer and sintering) are combined to predict the porosity and bond quality.

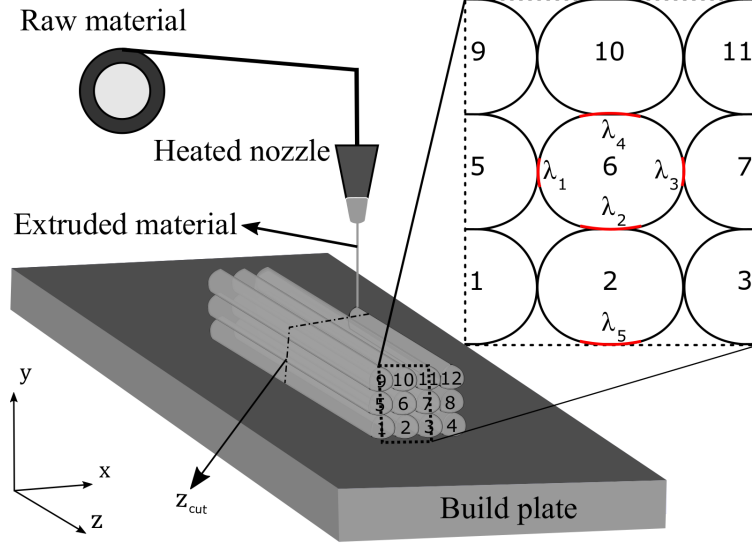


Figure 2.3: Schematic of FFF

2.5.1 FFF Temperature Modeling

Fused filament fabrication (FFF) (also known as fused deposition modeling or FDM[®]) is an AM technology based on material extrusion for manufacturing polymer and plastic parts. The continuous strand of material is pushed through a heated nozzle and deposited as a molten extruded thin filament onto the build plate in a predefined path to form the part in the desired shape. The schematic of the FFF process is shown in Fig. 2.3.

In FFF, each filament is subjected to the same heat transfer mechanism with different boundary conditions depending on thermal conditions such as environment, build plate and extrusion temperature, as well as the part geometry, material properties and deposition sequence. Recently, Costa et al. [73] developed an analytical solution for the transient heat transfer during the print process in FFF. The temperature prediction model considers conduction heat transfer with the build plate and adjacent filaments based on the fraction λ_i of filament perimeter that is in contact, and convection heat transfer with the environment. The deposition of filaments is modeled gradually by joining elementary lengths that are associated with a given deposition time. Axial and radial heat conduction are neglected due to the low thermal conductivity of polymers and small filament radius [73]. Thus, after these assumptions the mathematical energy balance for an elementary length dz can be written as:

$$\rho CA \frac{\partial T_m(z,t)}{\partial t} dz = - \left[h_{conv} A_m^{conv} (T_m(z,t) - T_S) + \sum_{i=1}^n h_i A_m^i (T_m(z,t) - T_m^i) \right], \quad (2.33)$$

where ρ , C , h_{conv} , and h_i are the material properties of the polymer and assumed to be temperature-independent. The parameter h_{conv} is the convective heat transfer coefficient, h_i represents the conduction heat transfer coefficient, ρ and C are the density and specific heat capacity of the material, T_m is the temperature at a specified cross-section $z = z_{cut}$ of the m -th filament ($m \in \{1, \dots, N\}$, where N is the total number of deposited filaments) at deposition time instant t , T_m^i represents the temperature of the adjacent filament or build plate at contact i ($i \in \{1, \dots, n\}$, where n is the total number of contact surfaces of a filament including the contact with the build plate) of the m -th filament or the build plate, T_S is the surrounding environment temperature, and A represents the cross-section area of a filament. $A_m^{conv} = P_m (1 - \sum_{i=1}^n \omega_m^i \lambda_i) dx$ is the area of the m -th filament that is in contact with the environment, $A_m^i = P_m \omega_m^i \lambda_i dx$ is the area of contact i for the m -th filament as shown in Fig. 2.4, where P_m is the filament perimeter, λ_i is the fraction of P_m that is in contact with another filament or with the build plate, and ω_m^i is a variable, which equals unity if the m -th filament has the i -th contact, and zero otherwise.

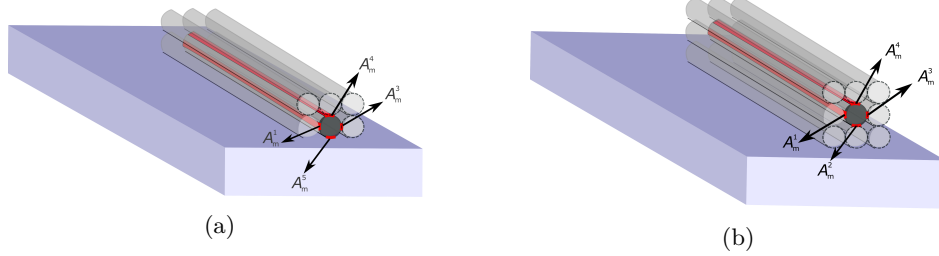


Figure 2.4: Possible contact areas of a filament in the: (a) first layer and (b) remaining layers

The analytical solution of Eq. (2.33) can be obtained using the characteristic polynomial method [74]:

$$T_m(z, t) = \phi_1 \exp \left[\frac{P_m \chi(\omega_m^1, \dots, \omega_m^n)}{\alpha A} (t - t_m(z)) \right] + \psi(\omega_m^1, \dots, \omega_m^n), \quad (2.34)$$

where $\phi_1 = T_m(t_m(z)) - \psi(\omega_m^1, \dots, \omega_m^n)$, $T_m(t_m(z))$ is the temperature of the m -th filament at instant $t_m(z)$ at which an elementary length z of the m -th filament is deposited and starts to cool down or contact with an adjacent filament or the build plate. A more detailed description of the derivation of temperature evolution can be found in Costa et al. [73]. The functions that are influenced by the

contacts χ and ψ are defined as:

$$\begin{aligned}\chi(\omega_m^1, \dots, \omega_m^n) &= h_{conv} \frac{A_m^{conv}}{P_m dz} + \sum_{i=1}^n h_i \frac{A_m^i}{P dz}, \\ \psi(\omega_m^1, \dots, \omega_m^n) &= \frac{h_{conv} \frac{A_m^{conv}}{P_m dz} T_S + \sum_{i=1}^n h_i \frac{A_m^i}{P_m dz} T_m^i}{\chi(\omega_m^1, \dots, \omega_m^n)}.\end{aligned}\quad (2.35)$$

2.5.2 Bond Formation Modeling

This subsection develops a new method to compute the rate of polymer sintering and the final mesostructure of the printed part. The sintering process is defined as the coalescence of particles, in which two particles of molten polymer form a homogeneous melt, under the action of surface tension [75]. The sintering process for amorphous polymers is driven by the surface tension force since the mechanism is considered a Newtonian viscous flow [76].

A Newtonian sintering model for polymers was initially developed by Frenkel et al. [77] to predict the rate of polymer sintering. Pokluda et al. [75] developed a closed-form equation to predict the bond formation between two spherical particles based on the work balance of viscous dissipation and surface tension. Bellehumeur et al. [78] applied the model proposed by Pokluda et al. [75] to FFF for predicting the sintering between adjacent filaments as a nonlinear function of time, temperature-dependent surface tension $\Gamma(T)$ and viscosity $\eta(T)$, and an initial particle radius. The model has limitations related to the mesostructure of the filaments; specifically, the geometry of the filaments is assumed constant during the printing process. Based on the above discussion, in this paper we propose a new sintering model, which considers realistic filament geometry (similar to the one proposed by Garzon et al. [79]) and also accounts for the change in the mesostructure of the filaments during the printing process.

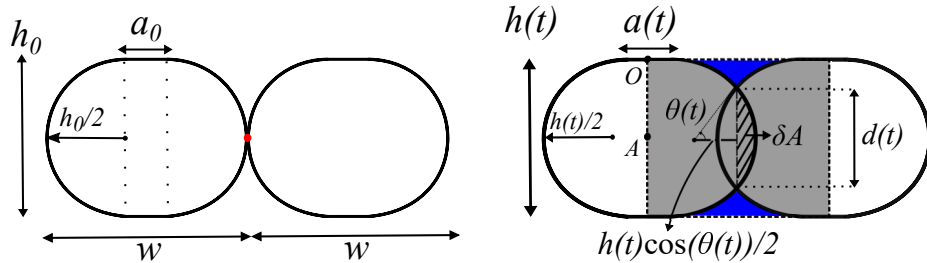


Figure 2.5: Evolution of neck diameter during sintering process

The sintering process is simulated by considering two symmetrical adjacent filaments. In this work, all filaments are assumed to undergo the same sintering process by neglecting the effect of location. The cross-section geometry of a filament is composed of a rectangle with an initial

width a_0 and two half circles with a radius of $h_0/2$ at initial time $t = 0$ as proposed by Garzon et al. [79]. At $t = 0$, adjacent filaments have one contact point between them (see Fig. 2.5). The width of the rectangle $a(t)$ evolves in time, together with $h(t)$ such that width of each filament $w = a_0 + h_0 = a(t) + h(t)\cos(\theta(t))$ stays constant. The layer height $h(t)$ is assumed to evolve in time based on the experimental observations, in which the average layer height for FFF parts has decreased more than the average width of filaments (see Fig. 2.6). During the sintering process, the width and length of each filament (w and L) are assumed to be constant. The coalescence between these two half circles forms the sintering angle $\theta(t)$ and neck diameter $d(t)$.

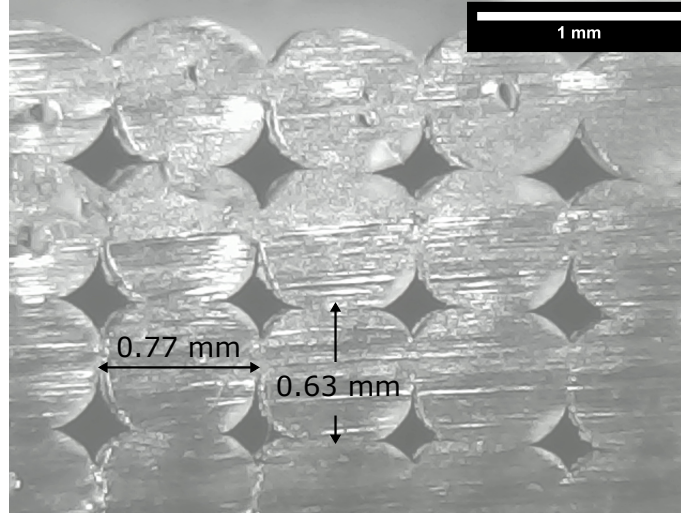


Figure 2.6: Cross-sectional geometry of an FFF part printed with an extrusion temperature 240°C, speed 42 mm/s, initial layer height 0.7 mm, filament width 0.8 mm, filament length 35 mm, 6 number of layers and 15 filaments per layer

In order to calculate the change in the filament geometry during the sintering process, the law of conservation of mass is expressed for two adjacent filaments that are assumed to have constant density:

$$2L \left(\pi \left(\frac{h_0}{2} \right)^2 + h_0 a_0 \right) = 2L \left(\pi \left(\frac{h(t)}{2} \right)^2 + h(t) a(t) - 2\delta A \right), \quad (2.36)$$

where $2\delta A = 2(h(t)/2)^2(\theta(t) - \sin(2\theta(t)))$ is the area of intersection between two filaments as shown in Fig. 2.5.

Thus, the evolution of layer height is obtained as

$$h(t) = \frac{-2w + \sqrt{Q}}{\pi - 2\theta + \sin(2\theta) - 4\cos(\theta)}, \quad (2.37)$$

where

$$\begin{aligned}
Q &= h_0^2 \pi^2 - 2h_0^2 \pi \theta + h_0^2 \pi \sin(2\theta) - 4h_0^2 \pi \cos(\theta) - 4h_0^2 \pi \\
&\quad + 8h_0^2 \theta - 4h_0^2 \sin(2\theta) + 16h_0^2 \cos(\theta) + 4h_0 \pi w \\
&\quad - 8h_0 \theta w + 4h_0 w \sin(2\theta) - 16h_0 w \cos(\theta) + 4w^2.
\end{aligned} \tag{2.38}$$

The work done by surface tension is defined as

$$W_S = -\Gamma(T) \frac{dS}{dt}, \tag{2.39}$$

where S is the total surface of the filaments that undergo sintering process and is given as

$$\begin{aligned}
S(t) &= h(t)^4 \left(-\theta + \frac{\sin(2\theta)}{2} \right) + h(t)^2 \pi + 2h(t)L\pi \\
&\quad - 4h(t)L\theta + 4h(t)a(t) + 4La(t).
\end{aligned} \tag{2.40}$$

Thus, applying the chain rule on Eq. (2.40), the work done by surface tension is obtained as

$$W_S = \Gamma(T)h(t) (2h(t)^3 \sin^2(\theta) + 4L) \theta'. \tag{2.41}$$

The work done by the viscous forces for a Newtonian fluid can be expressed as

$$W_v = \int \int \int_V \eta(T) \nabla u : (\nabla u + \nabla u^T) dV, \tag{2.42}$$

V being the volume of the sintering system, and ∇u the gradient of velocity and is expressed as

$$\nabla u = \begin{bmatrix} \dot{\epsilon}_1 & 0 & 0 \\ 0 & \dot{\epsilon}_2 & 0 \\ 0 & 0 & \dot{\epsilon}_3 \end{bmatrix}. \tag{2.43}$$

where $\dot{\epsilon}_i$ is the strain rate in i th direction.

Assuming the deformations in width and length are negligible w.r.t. deformations in height, $\dot{\epsilon}_2 = \dot{\epsilon}$ is approximated by

$$\dot{\epsilon} = \frac{\partial u_x(A)}{\partial x} \approx \frac{u_x(A) - u_x(O)}{OA_0} = \frac{\frac{d}{dt}h(t)}{h_0}. \tag{2.44}$$

Consequently, the work done by the viscous forces can be defined as follows

$$W_v = \int \int \int_V 2\eta(T)\dot{\epsilon}^2 dV \quad (2.45)$$

$$= \frac{2L\eta(T)h(t)^2 (h_0\pi - 4h_0 + 4w) \sin^2(\theta)}{h_0 \cos^2(\theta)} \theta''. \quad (2.46)$$

The evolution of the sintering angle $\theta(t)$ is then obtained by equating the work done by surface tension and the viscous forces under the assumption that θ' is always positive [75]:

$$\frac{d\theta(t)}{dt} = \frac{\Gamma(T)h_0 \left(h(t)^3 + \frac{2L}{\sin^2(\theta(t))} \right) \cos^2(\theta)}{Lh(t)\eta(T)(h_0\pi - 4h_0 + 4w)}. \quad (2.47)$$

The evolution of neck diameter $d(t)$ with time is then computed as

$$d(t) = h(t) \sin(\theta(t)), \quad (2.48)$$

The porosity can be expressed using the geometry of the mesostructure shown in Fig. 2.5, where the shaded grey region is the filled area and blue region is the void area. Thus, the evolution of porosity $\phi(t)$ is calculated as

$$\phi(t) = \frac{h^2 \cos(\theta) - \left(\frac{h}{2}\right)^2 [\pi - (2\theta - \sin(2\theta))]}{h^2 \cos(\theta) + ha}. \quad (2.49)$$

The proposed modeling approach is an improvement upon the previous work on sintering models by Pokluda et al. [75] and Gurralla et al. [80], because (i) it considers a realistic filament geometry based on our experiments, and (ii) it accounts for changes in the filament geometry during the printing process. This proposed methodology still makes assumptions about the bonding process and sintering is not the only physical phenomenon that takes place during the bond formation. Thus, physics-informed machine learning (PIML) models are studied in Chapter 3 to further enhance the prediction accuracy.

2.6 Summary

This chapter described the physics-based models and machine learning algorithms used later in the dissertation. A Bayesian neural network (BNN), which is used in Chapter 7, is introduced as a generic framework to quantify model prediction uncertainty. The basic concepts of uncertainty quantification (UQ) techniques are reviewed in Section 2.3. Global sensitivity analysis introduced in

Section 2.4 is used in Chapters 4 and 7 to quantify the relative contribution of each random variable on the system-level variability. Bayesian calibration techniques are used in the later chapters of the dissertation. In Chapter 3, the physics models are used to generate training data, then deep neural networks are trained to predict the part quality. The physics-based and ML models are integrated in an innovative manner to better capture the dynamics of the AM process. In Chapter 5, adaptive surrogate modeling techniques are investigated. In Chapter 4, Gaussian process (GP) and deep neural network (DNN) are used to develop physics-informed machine learning (PIML) models to predict the output quantity of interest (QoI), and the uncertainties in these models are included in global sensitivity analysis. In Chapter 6, multi-level information fusion for model calibration is investigated using the available measurement data. In Chapter 7, some of the UQ techniques are used to study the process parameter optimization under uncertainty.

CHAPTER 3

Machine Learning using both Physics Knowledge and Experimental Data^{1,2}

3.1 Introduction

Physics-based models do not require large amounts of data, but are generally limited by their computational complexity or incomplete physics. In contrast, ML models appear promising for complex systems that are not fully understood or represented with simplified relationships, given adequate quality and quantity of data. However, ML models represent the complex physics without taking into account any physical laws and thus can produce results that are inconsistent with physical laws. This chapter investigates different strategies to enhance the experimental data-driven ML models by incorporating the physics knowledge, and illustrates them for additive manufacturing (AM).

Achieving the desired material properties and product quality in AM processes has been studied using trial-and-error experiments as well as process models (either physics-based or ML models). In a trial-and-error approach, the AM process is repeated multiple times with different process parameter combinations to achieve the desired microstructure and properties of the manufactured parts; this is expensive and time-consuming. Moreover, this trial-and-error approach needs to be implemented every time a new design needs to be manufactured. Therefore, in recent years, research efforts have focused on model-based methods for optimizing the AM process parameters.

Several physics-based models have been developed depending on the AM process category and the quantity of interest (QoI) [81]. Costa et al. [73] proposed an analytical solution for transient heat transfer during the printing process in fused filament fabrication (FFF). Different models have been proposed in the literature to study polymer sintering [75, 77, 79, 82]. Many of these models are parametric representations of complex physical processes based on various approximations. The parameters of such physics-based models, as well as the model errors need to be calibrated for each AM process using available observation data to reduce the uncertainty in the model predictions [26, 32, 51]. Due to the complex physics of the AM process, a different model is needed for each sub-stage or phenomenon in the manufacturing process in order to accurately predict the QoI. Further, physics models with reasonable fidelity and accuracy require tremendous computational effort; as a result, the use of physics-based modeling in AM of realistic products has been challenging and limited.

¹Adapted with permission from: Kapusuzoglu, B., & Mahadevan, S., "Physics-Informed and Hybrid Machine Learning in Additive Manufacturing: Application to Fused Filament Fabrication," JOM, vol. 72, no. 12, 2020..

²© 2020 by The Minerals, Metals & Materials Society

Recently, several studies have used the available AM experimental data to build black-box ML models. In addition, with increased computing power, deep learning has become a prominent tool for solving classification and regression problems. In the context of AM, Khanzadeh et al. [83] compared supervised machine learning approaches to classify melt pools to predict porosity. Artificial neural network has been used to predict the geometry of a single bead in wire and arc additive manufacturing from the wire-feed rate and travel speed [28]. Kwon et al. [84] investigated the convolutional neural network (CNN) to predict laser power from melt pool images. Zhang et al. [85] used a CNN model to perform in-process porosity monitoring of laser-based AM processes.

In the context of physics-informed machine learning Karpatne et al. [86] proposed the combined use of physics-based and ML models to achieve more accurate and physically consistent predictions by leveraging the advantages of each method. In order to make ML models consistent with physical laws, Karpatne et al. [86] incorporated physical constraints into the loss function of ML models. Another method that is also implemented by Karpatne et al. [86] to combine physics-based and ML models is to use the physics-based model outputs as additional inputs in an ML model along with other inputs. Jia et al. [87] used synthetic data generated by executing physics-based models for multiple input combinations to pre-train the ML model in order to leverage the knowledge embedded in physics-based models. These general ideas have been explored in multiple engineering applications, such as geoscience, fluid dynamics, and thermodynamics, and this paper investigates these ideas for additive manufacturing.

The investigation of physics-informed machine learning (PIML) models for the prediction of AM part quality is the first objective of this dissertation. Therefore, this chapter investigates several physics-informed and hybrid machine learning strategies that incorporate physics knowledge in experimental data-driven deep learning models for predicting the bond quality and porosity of FFF parts, thus helping the ML model to produce more accurate and physically meaningful results. Fused filament fabrication (FFF), an extrusion-based deposition technique as shown in Fig. 2.3, is a widely used AM process. The bond quality between adjacent filaments and layers strongly affects the mechanical properties of FFF-produced parts. Thus, it is important to predict the bond formation, and therefore the mechanical properties of the manufactured part accurately. Although physics-based models predicting the temperature evolution, bond formation and mesostructure evolution of FFF parts are based on physical laws, they introduce bias due to incomplete representation of the complex physical process by approximating the reality. In addition, these models contain a significant number of model parameters that need to be calibrated using experimental data [26]. On the other hand, ML models are not aware of physical laws, which may result in physically inconsistent model

predictions. However, they can extract complex physical relationships from available data. Thus, physics-based models and ML models can be integrated in an innovative manner to better capture the dynamics of the AM process.

In this chapter, the goal is to enhance the experimental data-driven ML models for AM by incorporating the physics knowledge, thus helping the ML model to produce more accurate and physically meaningful results. Three types of strategies are explored to incorporate physics constraints and multi-physics FFF simulation results into a deep neural network (DNN), thus ensuring consistency with physical laws: (1) incorporate physics constraints within the loss function of the DNN, (2) use physics model outputs as additional inputs to the DNN model, and (3) pre-train a DNN model with physics model input-output and then update it with experimental data. These strategies help to enforce a physically consistent relationship between bond quality and tensile strength, thus making porosity predictions physically meaningful. Eight different combinations of the above three strategies are explored, and their performance in porosity and bond quality prediction of the FFF-produced parts is examined.

In summary, several physics-informed and hybrid machine learning models are developed for porosity and bond quality prediction of FFF parts using physics constraints, physics-based model, and experimental data, using the three strategies mentioned above. Next, an enhanced physics-based model is developed to account for realistic filament geometry and the change of geometry during the printing process (see Section 2.5.2). Lastly, the proposed models are trained and evaluated using laboratory experiments where FFF parts are printed with varying input conditions, and data is collected to measure the quality characteristics (bond quality, porosity) of the parts.

3.2 Proposed Methodology

This section develops the proposed physics-informed machine learning (PIML) approaches to predict the bond formation and mesostructure in FFF-produced parts. The methodology is applicable to any AM process with corresponding data and physical laws for the prediction QoI. The three components of the methodology are: (a) Physics-based models, (b) Experiments, and (c) Construction of PIML models.

3.2.1 Physics-Informed Machine Learning (PIML) for Additive Manufacturing

Although physics-based models predicting the temperature evolution, bond formation and mesostructure evolution of FFF parts are based on physical laws, they introduce bias due to incomplete representation of the complex physical process by approximating the reality. In addition, these models

contain a significant number of model parameters that need to be calibrated using experimental data [26]. On the other hand, ML models are not aware of physical laws, which may result in physically inconsistent model predictions. However, they can extract complex physical relationships from available data. Thus, physics-based models and ML models can be integrated in an innovative manner to better capture the dynamics of the AM process.

In PIML models, physics knowledge and data are sought to be integrated in a synergistic manner by leveraging the complementary strengths of both models [86]. Thus, the goal is to improve the predictions beyond that of physics-based models or ML models alone by coupling physics-based models with ML models. In the following, three different strategies to combine physics knowledge and ML models are pursued: (1) incorporate physics constraints within the loss function of the DNN, (2) use physics model outputs as additional inputs to the DNN model, and (3) pre-train a DNN model with physics model input-output and then update it with experimental data.

3.2.1.1 Physics-Informed Loss Functions

A direct strategy to improve ML model predictions is by including physics-based loss functions [86]. Consider a PIML model with inputs X and outputs \hat{Y} trained using physical laws that are incorporated as constraints into the loss function:

$$\mathcal{L} = \mathcal{L}_{\text{DNN}}(Y, \hat{Y}) + \sum_{k=1}^M \lambda_{\text{phy},k} \mathcal{L}_{\text{phy},k}(\hat{Y}), \quad (3.1)$$

where \mathcal{L}_{DNN} is the regular training loss of a DNN that evaluates a supervised error (e.g., root mean squared error (RMSE)); $\mathcal{L}_{\text{DNN}}(Y, \hat{Y}) = \sqrt{\sum_{i=1}^n (Y_i - \hat{Y}_i)^2 / n}$, which measures how far off the predictions \hat{Y} are from the observations Y for the n training samples, and $\mathcal{L}_{\text{phy},k}$ is the k -th physics-based loss function, whose contribution is controlled by a hyperparameter $\lambda_{\text{phy},k}$ and M is the total number of physics-based loss functions. The inclusion of $\mathcal{L}_{\text{phy},k}$ ensures physically consistent model predictions (the second term of Eq. (3.1) means physical inconsistency) and can decrease the generalization error even when there is a small amount of training data [86]. In addition, $\mathcal{L}_{\text{phy},k}$ does not require experimental observations; the data obtained from the physics model is used to evaluate physics-based loss functions.

In this work, we enforce five different physics-based loss functions (i.e., five separate physical relationships, $\mathcal{L}_{\text{phy},k}(\hat{Y})$, where $k = \{1, 2, 3, 4, 5\}$ and $\hat{Y} = (\hat{Y}_1, \hat{Y}_2)$ are the overall dimensionless neck diameter (i.e., overall bond quality) and porosity predictions of FFF parts, respectively). These loss

functions are defined as follows:

$$\begin{aligned}
\mathcal{L}_{\text{phy},1}(\hat{Y}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(-\hat{Y}_{1,i}), \\
\mathcal{L}_{\text{phy},2}(\hat{Y}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(\hat{Y}_{1,i} - d_{\text{max}}), \\
\mathcal{L}_{\text{phy},3}(\hat{Y}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(-\hat{Y}_{2,i}), \\
\mathcal{L}_{\text{phy},4}(\hat{Y}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(\hat{Y}_{2,i} - \phi_{0,i}), \\
\mathcal{L}_{\text{phy},5}(\hat{Y}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(\Delta_i),
\end{aligned} \tag{3.2}$$

where the first four loss functions consider the physical violations related to the overall dimensionless neck diameter and porosity across N samples and the fifth loss function represents the physical relationship between the mechanical properties and neck diameter. The physical inconsistencies in the model predictions are evaluated using these physics-based loss functions. In the first and third loss functions, negative values of neck diameter and porosity are treated as physical violations. The second loss function evaluates physically inconsistent dimensionless neck diameter predictions which are greater than the maximum dimensionless neck diameter $d_{\text{max}} = 1$. The fourth loss function penalizes the model when porosity predictions $\hat{Y}_{2,i}$ are greater than the initial porosity $\phi_{0,i}$ of i th part. This is based on the physics knowledge that the total void area decreases as the sintering process takes place. The fifth physics-based loss function exploits the monotonic relationship between bond quality and tensile strength of FFF-produced parts. This loss function is constructed by computing the difference in the sorted dimensionless neck diameter predictions, $\hat{Y}_{1,\text{sorted}}$, and dimensionless neck diameter predictions corresponding to sorted tensile strength estimates ($\sigma_{TS}(\hat{Y}_{2,\text{sorted}}, \xi)$), $\hat{Y}'_{1,i}$, i.e., $\Delta_i = \hat{Y}_{1,\text{sorted},i} - \hat{Y}'_{1,i}$. The maximum stress for longitudinal raster orientation, $\sigma_{TS}(\hat{Y}_2, \xi)$ with $\xi = \{\sigma_{01}, \sigma_{02}, C_\sigma\}$ being the material parameters, is computed according to the analytical expression proposed in Garzon et al. [79] using the porosity predictions:

$$\sigma_{TS} = \sigma_{01} \left[\exp\left((1 - \hat{Y}_2)^{C_{\sigma n_1}}\right) - \hat{Y}_2 \right] + \sigma_{02}(1 - \hat{Y}_2), \tag{3.3}$$

where n_l is the number of layers of FFF parts. The tensile strength is constrained when porosity is equal to 0, i.e., $\sigma_{TS} = \sigma_{01}e + \sigma_{02}$. The overall average neck diameter and tensile strength are positively correlated, and tensile strength increases monotonically with neck diameter. Whereas, porosity and tensile strength are negatively correlated (as are porosity and neck diameter). More specifically,

the model predictions $(\hat{Y}_{1,i}, \hat{Y}_{2,i})$ and $(\hat{Y}_{1,i+1}, \hat{Y}_{2,i+1})$ corresponding to i th and $(i+1)$ th FFF parts can be used to estimate $\sigma_{TS,i}$ and $\sigma_{TS,i+1}$. If $\sigma_{TS,i+1}$ is greater than $\sigma_{TS,i}$ — meaning $(i+1)$ th part has less voids than i th part ($\hat{Y}_{2,i+1} < \hat{Y}_{2,i}$)—then $\hat{Y}_{1,i+1}$ should be greater than $\hat{Y}_{1,i}$ as well by exploiting a key monotonic physical relationship between porosity and tensile strength of FFF-produced parts. Thus, with the inclusion of these physics-based penalty functions, the neck diameter and porosity predictions are ensured to be physically meaningful.

3.2.1.2 Physics Model Output as Additional ML Model Input

A physics-based model $f^{\text{phy}} : X \rightarrow \hat{Y}^{\text{phy}}$ can be used to predict the QoI, where \hat{Y}^{phy} are predicted estimates of the true response of the system Y . A straightforward approach to combine physics-based and ML models is to use physics model output \hat{Y}^{phy} (at the experimental inputs) as additional input along with inputs X ; i.e., $f^{\text{hyb}} : X^{\text{hyb}} = [X, \hat{Y}^{\text{phy}}] \rightarrow \hat{Y}^{\text{hyb}}$.

Adding the physics output as an extra input to the DNN model (which is trained using experimental data) is information fusion, where the physics model is an additional source of information that is consistent with physics (i.e., when the physics model satisfies the constraints mentioned in the previous subsection). The resulting DNN model can be thought of as a hybrid model that uses the experimental data to correct the output of the physics model which is an incomplete representation of the actual physics.

3.2.1.3 Pre-trained PIML Model

In AM, especially in the FFF process with not a high-quality printer, parts have significant variability in quality. There is also uncertainty in measurement and lack of data due to the high cost associated with conducting experiments. Thus, data of adequate quality and quantity is important for good quality model predictions in AM.

In order to leverage the complex physical knowledge inherent in the physics-based models, synthetic data can be generated for multiple input combinations using physics-based models. The synthetic data can be used to train a ML model, which is used as the initial model to be updated with experimental data. The transfer of physical knowledge using a pre-trained ML model can prevent poor initialization due to lack of knowledge of initial choice of ML model parameters prior to training. This allows the pre-trained ML model to be fine-tuned even with limited observed data. In addition, it has been shown that using synthetic data from even imperfect physics models with uncalibrated model parameters can still reduce the amount of experimental training data needed [87].

More importantly, the pre-training can use a large amount of training data (with multiple input parameter combinations) over a wide range of values, which is not possible in experiments that could be expensive; as a result, the pre-training may help the eventual ML model to have wider generalization beyond experimental data. This is also an important distinction of the pre-training strategy from the second strategy. Both strategies use the physics model, but in the second strategy, the physics model is only used to provide outputs corresponding to the experimental inputs, whereas in the current pre-training strategy, the physics model is used to provide outputs corresponding to a much larger set of inputs. In the numerical example in Section 3.3, the pre-training strategy exercises the physics model over 1525 input combinations, whereas the second strategy above only employs the physics model over 39 experimental input combinations. However, the advantage of the pre-training strategy in using a larger input data set (for physics model runs) compared to the experiments becomes limited if the physics model is computationally expensive.

In this work, the ML model is pre-trained using the outputs of an uncalibrated coupled multi-physics model (i.e., neck diameter and porosity). Further, the transfer of learned physical knowledge is shown to be valuable even when the input parameters of the synthetic data generated are quite different than the experimental observations. Once the ML model is pre-trained, it is fine-tuned using limited experimental observations. This helps to learn a 3D printer-specific physical process faster and with less samples.

The three proposed strategies to predict the QoIs are shown in Fig. 3.1. Figure 3.1(a) shows the first method, where the physical knowledge is included through constraints within the loss function of a DNN trained with experimental data. Figure 3.1(b) shows the second method, where the outputs of the physics model are additional inputs to the DNN model. Figure 3.1(c) shows the third method, where a DNN model is pre-trained with data generated using the physics-based model and then updated using experimental data. The proposed PIML strategies can be applied to any AM process by leveraging the physical constraints or physics-based models.

3.2.1.4 Combination of PIML Strategies

Based on the proposed three strategies to incorporate physics knowledge into the ML model, eight separate ML models can be constructed:

- | | |
|--|--|
| 1. DNN | 4. DNN^{upd} |
| 2. $\text{DNN}^{\mathcal{L}_{\text{phy}}}$ | 5. $\text{DNN}^{\text{hyb}, \mathcal{L}_{\text{phy}}}$ |
| 3. DNN^{hyb} | 6. $\text{DNN}^{\text{upd}, \text{hyb}}$ |

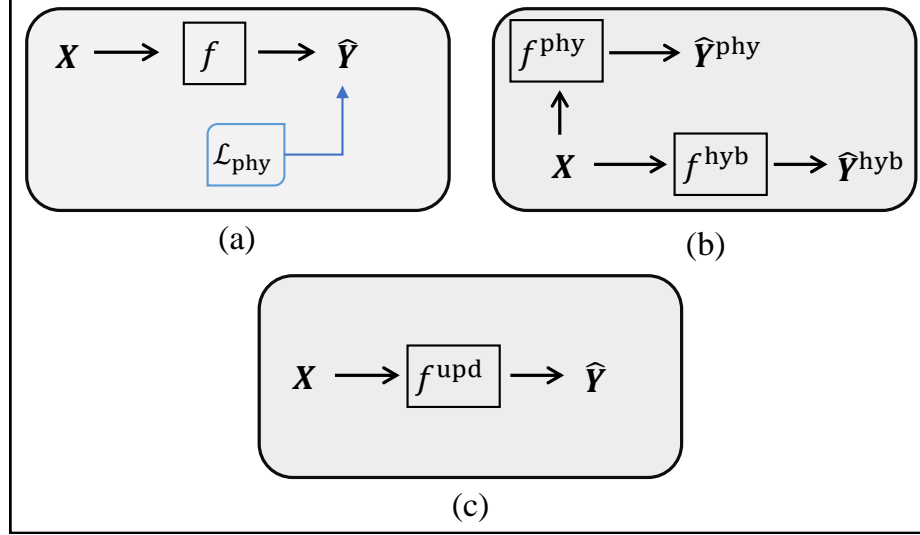


Figure 3.1: PIML strategies: (a) incorporate physics constraints within the loss function of the DNN, (b) use physics model outputs as additional inputs to the DNN model, and (c) pre-training a DNN model with physics model input-output and updating it with experimental data

7. $\text{DNN}^{\text{upd}, \mathcal{L}_{\text{phy}}}$

8. $\text{DNN}^{\text{upd,hyb}, \mathcal{L}_{\text{phy}}}$

In model 1, a deep neural network DNN is trained using only experimental data. The inputs X for this basic DNN model are the process parameters, printer extrusion temperature, speed, layer height, filament width, length, number of layers, and number of filaments per layer; and the outputs are overall dimensionless neck diameter and porosity. These inputs and outputs are the same as those used in the physics-based model f^{phy} . Model 2 ($\text{DNN}^{\mathcal{L}_{\text{phy}}}$) pursues the first strategy: physical knowledge related to the FFF process is included through constraints within the loss function of the DNN as shown in Eq. (3.1). Model 3 pursues the second strategy: a hybrid physics-based neural network DNN^{hyb} is trained using the outputs \hat{Y}^{phy} of f^{phy} as extra inputs in addition to X , i.e., $X^{\text{hyb}} = [X, \hat{Y}^{\text{phy}}]$. Model 4 (DNN^{upd}) pursues the third strategy, where the weights and biases (model parameters) of all the layers excluding the input layer of the pre-trained network f^{pre} (which is trained with the coupled multi-physics model input-output described in Section 3.2.1.3) are used as initial parameters for the DNN model and these parameters are updated with experimental data.

The architecture of the pre-trained model and the updated models (Model 4, 6, 7, and 8) is the same (except the input layer which changes with different numbers of inputs). The rest of the models (5-8) represent the combinations of the three strategies. Models 5, 6, and 7 each combine any two of the three strategies, whereas model 8 combines all three strategies. Model 5 ($\text{DNN}^{\text{hyb}, \mathcal{L}_{\text{phy}}}$) combines the use of physics model outputs as additional inputs and the incorporation of physics

constraints \mathcal{L}_{phy} within the loss function of the DNN. Model 6 ($\text{DNN}^{\text{upd,hyb}}$) combines second and third strategies, where the optimized model parameters of f^{pre} are used as the initial values and the outputs \hat{Y}^{phy} of f^{phy} are included as additional inputs. $\text{DNN}^{\text{upd,hyb}}$ has the same number of inputs as DNN^{hyb} (i.e., both include the physics model output as additional input) and uses the optimized model parameters of all the layers excluding the input layer of f^{pre} as initial parameters before updating with experimental data. Model 7 ($\text{DNN}^{\text{upd},\mathcal{L}_{\text{phy}}}$) combines first and third strategies. The model parameters obtained from f^{pre} are updated using the experimental data by minimizing the augmented loss function shown in Eq. 3.1. Model 8 ($\text{DNN}^{\text{upd,hyb},\mathcal{L}_{\text{phy}}}$) combines the use of physics model outputs as additional inputs to the updated DNN model DNN^{upd} (which results in $\text{DNN}^{\text{upd,hyb}}$) and the physics constraints \mathcal{L}_{phy} are incorporated within the loss function of $\text{DNN}^{\text{upd,hyb}}$.

3.3 Numerical Example

In this section we demonstrate the implementation of the proposed methodology to FFF-produced parts, and investigate the performance of eight PIML models described in Section 3.2.1.4. The results show that the proposed PIML models are capable of achieving physically meaningful and accurate model predictions, and require a smaller number of experiments.

3.3.1 Problem Setup

First, data is collected from laboratory experiments in order to build the prediction models for the QoI using ML techniques. The shape of the part is conceptualized, a CAD model is visualized, and then sliced in a slicing software using the defined FFF process parameters and printing path.

The print quality depends on the adhesion of the first layer with the build plate [26]. Thus, several measures are needed to ensure proper adhesion. For example, the printing environment is modified by adding an enclosure to the 3D printer to isolate the printing environment from external effects. Kapton tape is used on the glass build plate to enhance the adhesion of Acrylonitrile butadiene styrene (ABS) with the build plate. After these modifications, the part is printed and then measured with appropriate monitoring techniques.

A commercial material Ultimaker Black ABS was extruded from an Ultimaker 2 Extended+ 3D printer to manufacture parts with unidirectionally aligned filaments. Using Latin hypercube sampling, 20 sets of process parameters are generated. The ranges considered for the variables are printer extrusion temperature T_e : (210°C - 260°C), and extrusion speed S_e : (15 mm/s - 46 mm/s). Since the values of material properties ξ do not affect the outcome of $\mathcal{L}_{\text{phy},5}(\hat{Y})$, the values calibrated by Garzon et al. [79] are used. All specimens were sectioned at the midpoint $z_{\text{cut}} = L/2$ (since

the statistical properties of QoIs along the length of the specimens were constant) to analyze the mesostructural feature of interest with the use of microscopy images processed through the ImageJ software [88]. The collected experimental data is subsequently used to create DNN prediction models.

For the physics-based models, the surface tension of ABS P400 at 240°C is assumed 0.029 N/m as reported by Bellehumeur et al. [78] with a temperature dependence $\Delta\Gamma(T)/\Delta T = -\gamma \text{ N/m} \cdot \text{K}$, where the model parameter $\gamma = 0.00345$. The temperature dependent material viscosity $\eta(T)$ is given by $\eta(T) = \eta_r \exp[-\beta(T - T_r)]$, where the material viscosity at the reference temperature ($T_r = 240^\circ\text{C}$) is $\eta(T = T_r) = \eta_r = 5100 \text{ Pa} \cdot \text{s}$, $\beta = 0.056$ as selected by Sun et al. [89]. The build plate temperature was constant and set to 110°C.

3.3.2 Model Training and Prediction

The eight DNN models were implemented using the Keras package [90] with Tensorflow backend. The pre-trained model, f^{pre} , is first trained with physics model input-output data consisting of 1525 input parameter combinations over a range of experimental values, i.e., ($210^\circ\text{C} \leq T_e \leq 260^\circ\text{C}$, $15 \text{ mm/s} \leq S_e \leq 46 \text{ mm/s}$), and then updated for different combinations of the proposed strategies using observed data. (Note that in contrast only 39 physical experiments with 20 unique input parameter combinations are available, see Fig. 3.2). The input data of the training and test sets are normalized prior to the training of the DNN models (the output quantities are dimensionless and between 0 and 1), and the hyperparameters of these models are tuned with grid search ($\lambda_{\text{phy}} = 0.3, 0.3, 0.15, 0.15, 0.008$). Fully-connected DNN models with 2 hidden layers and 10 neurons in each hidden layer are constructed and the weights of all neurons in these models are uniformly randomly initialized between 0 and 1. L1 and L2 regularizers are used as a penalty term to avoid overfitting. The Rectified Linear Unit (ReLU) activation function and Adam optimizer are used to perform stochastic gradient descent in learning the model parameters.

The number of epochs for the convergence of training is approximately the same for each model except f^{pre} , which converges in 40 epochs. The computation time for training of each model is on average 15 sec using a desktop computer (Intel® Xeon® CPU E5-1660 v4@3.20GHz with 32 GB RAM and GPU NVIDIA Quadro K620 with 2 GB).

3.3.3 Model Performance

In order to measure the accuracy of the trained DNN models, the model predictions of the test data are compared with the observed overall neck diameters and porosity. Each model is trained 30 times and compared against the FFF experimental data not used for training (i.e., data from

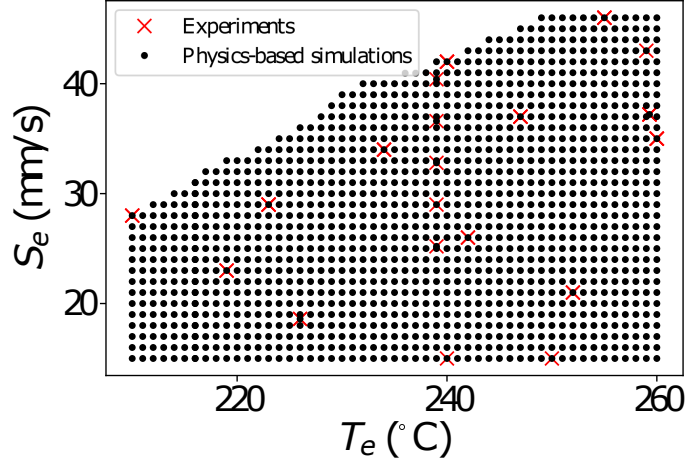


Figure 3.2: Design of experiments for physics-based simulations and experiments

19 parts) to evaluate the mean and standard deviation of RMSE and physical inconsistency. The effect of the training data size on the RMSE of different models is shown in Table 3.1. Here, the models are trained with 4, 6, 8, 10, and 20 experimental data points (i.e., 20%, 30%, 40%, 50%, and 100% of the available training data). The RMSE values of the coupled-physics model based on the sintering model developed by Gurrala et al. [80] and the new proposed one are 0.173 and 0.112, respectively, which are larger than the ML models due to the approximations used to represent the FFF process and bias in the model. The mean RMSE value of f^{pre} , which is trained with physics model input-output data consisting of 1525 input parameter combinations is 0.362. The RMSE value of f^{pre} is greater than the RMSE of f^{phy} because the pre-trained model is an approximation of the physics model, which causes uncertainty and bias. The RMSE of the basic DNN model is about 0.025 when we use all the parts in the training set. The RMSE of the basic DNN model is better than f^{pre} and f^{phy} because experimental data is directly used as the training data for the basic DNN model, whereas f^{pre} and f^{phy} use the approximate physics model to generate either pre-training data or additional input to the ML model.

The physics-based loss functions allow $\text{DNN}^{\mathcal{L}_{\text{phy}}}$ and $\text{DNN}^{\text{hyb}, \mathcal{L}_{\text{phy}}}$ to achieve physically meaningful results with a lower value of average RMSE than DNN^{hyb} and improve the generalization performance. The DNN^{hyb} model performs similar to DNN, which shows that the physics model outputs do not improve the learning process significantly. The DNN^{upd} and $\text{DNN}^{\text{upd}, \text{hyb}}$ models achieve a similar performance improvement as the DNN models that include physics-based loss functions. However, the models without physics constraints produce physically inconsistent results as shown in Table 3.1. The results show that pre-training the PIML model improves the performance, and the improvement

Table 3.1: Effect of different amounts of training data on the RMSE of different ML models

Model	20%	30%	40%	50%	100%	Mean Physical Inconsistency
f^{phy}	-	-	-	-	0.112	0.000
1. DNN	0.101(± 0.022)	0.084(± 0.018)	0.044(± 0.017)	0.028(± 0.004)	0.025(± 0.005)	0.201
2. DNN $^{\mathcal{L}_{\text{phy}}}$	0.055(± 0.017)	0.050(± 0.011)	0.025(± 0.005)	0.024(± 0.007)	0.021(± 0.005)	0.000
3. DNN $^{\text{hyb}}$	0.098(± 0.021)	0.078(± 0.024)	0.044(± 0.016)	0.028(± 0.007)	0.024(± 0.005)	0.195
4. DNN $^{\text{upd}}$	0.058(± 0.012)	0.049(± 0.011)	0.026(± 0.005)	0.025(± 0.005)	0.020(± 0.005)	0.133
5. DNN $^{\text{hyb}, \mathcal{L}_{\text{phy}}}$	0.059(± 0.018)	0.041(± 0.012)	0.026(± 0.004)	0.023(± 0.004)	0.020(± 0.005)	0.000
6. DNN $^{\text{upd}, \text{hyb}}$	0.057(± 0.014)	0.047(± 0.011)	0.026(± 0.003)	0.026(± 0.005)	0.024(± 0.006)	0.134
7. DNN $^{\text{upd}, \mathcal{L}_{\text{phy}}}$	0.054(± 0.011)	0.045(± 0.013)	0.026(± 0.003)	0.025(± 0.005)	0.021(± 0.005)	0.000
8. DNN $^{\text{upd}, \text{hyb}, \mathcal{L}_{\text{phy}}}$	0.053(± 0.013)	0.040(± 0.010)	0.025(± 0.004)	0.023(± 0.004)	0.018(± 0.003)	0.000

is relatively larger as the amount of observed data gets smaller. Additionally, the models that are pre-trained (DNN $^{\text{upd}}$ and DNN $^{\text{upd}, \text{hyb}}$) reach a physically more consistent initialization when they are updated with experimental data even without using physics constraints, compared to models that are not pre-trained, DNN and DNN $^{\text{hyb}}$. The combination of all strategies (DNN $^{\text{upd}, \text{hyb}, \mathcal{L}_{\text{phy}}}$) allows the model to get closest to the ground truth.

The prediction accuracy (w.r.t. neck diameter and porosity) of different models trained with 100% of the experimental data is shown in Fig. 3.3. The x and y-axis represent the physical inconsistency and the mean and standard deviation of RMSE, respectively. Figure 3.3 shows that models with physics-based loss functions produce physically consistent results. The incorporation of the physics knowledge using either the first or third strategy enables the models 2, 4-8 to generalize to configurations unseen in the training set.

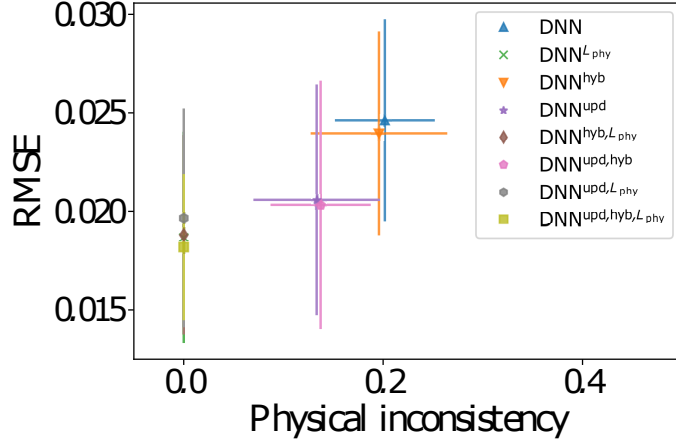


Figure 3.3: Performance and physical inconsistency of proposed models

In order to further analyze the improvement in model predictions, the predicted dimensionless overall neck diameter and porosity for the test set that comprises 19 FFF parts are visualized in

Fig. 3.4. Porosity predictions of model 1 (DNN) have large physical inconsistency (Fig. 3.4(a)). For instance, the blue triangle with the lowest porosity prediction has a negative value (i.e., $(\hat{Y}_1, \hat{Y}_2) = (0.52, -0.025)$). Model 2 ($\text{DNN}^{\mathcal{L}_{\text{phy}}}$) predictions are physically consistent due to enforced physics constraints, but model 3 (DNN^{hyb}) predictions also do not follow a monotonic decreasing relationship. Model 4 and 6 (DNN^{upd} and $\text{DNN}^{\text{upd,hyb}}$) have some physically inconsistent predictions. Figure 3.4(b) shows that $\text{DNN}^{\text{hyb},\mathcal{L}_{\text{phy}}}$, $\text{DNN}^{\text{upd},\mathcal{L}_{\text{phy}}}$ and $\text{DNN}^{\text{upd,hyb},\mathcal{L}_{\text{phy}}}$ produce physically consistent model predictions, i.e., porosity and neck diameter predictions follow a monotonically decreasing relationship.

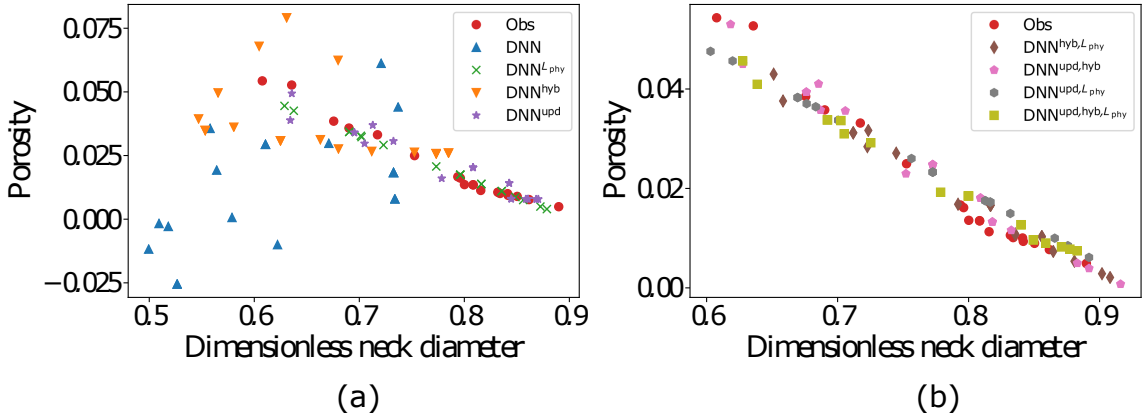


Figure 3.4: Comparison of model prediction with test data: (a) for the first 4 models, and (b) for the last 4 models

3.4 Summary

In this chapter, three strategies for physics-informed machine learning (PIML) are investigated for predicting the quality-related metrics of FFF-produced parts. First, a physics-based sintering model is developed to predict the overall average neck diameter and porosity of FFF parts using the temperature evolution of filaments, material properties, part geometry, and process parameters as inputs. The developed sintering model offers two improvements over existing models: (i) consideration of realistic filament geometry, and (ii) allowing the filament geometry to change during the printing process. Next, several PIML models are developed to predict the bond quality and porosity of FFF parts by leveraging three strategies for incorporating physics knowledge into the DNN model: (1) physics-based loss functions, (2) using the outputs of the coupled multi-physics model as additional inputs to the DNN, and (3) pre-training a DNN with data generated using physics-based model and then updating it with experimental data. The physics-based loss functions exploit the relationship between bond quality and tensile strength of the FFF parts.

The numerical results show that the incorporation of physics knowledge not only improves the prediction accuracy while producing physically meaningful results but also allows accurate model predictions even with smaller amounts of experimental data. Thus, the proposed approach helps to fill the physics knowledge gap in the ML model while leveraging the capability of ML to extract complex process-material-geometry relationships in AM, and correcting for the approximation in the physics-based model.

CHAPTER 4

Sensitivity Analysis with Hybrid Machine Learning Models^{1,2}

4.1 Introduction

When computational models (either physics-based or data-driven) are used for the sensitivity analysis of engineering systems, the sensitivity estimate is affected by the accuracy and uncertainty of the model. In high dimensional problems, even the use of a surrogate model for global sensitivity analysis (GSA), which repeatedly executes the code by suppressing some variables and running through the range of other variables, may be computationally demanding since the number of executions of the code increases rapidly with the number of inputs [91–94]. This objective considers GSA for situations where both a physics-based model and experimental observations are available, and investigates physics-informed machine learning (PIML) strategies to effectively combine the two sources of information in order to maximize the accuracy of the sensitivity estimate even with smaller amounts of experimental data.

The uncertainty sources affecting model prediction include (a) epistemic uncertainty due to lack of knowledge (arising from either data or model inadequacies), and (b) aleatory uncertainty due to the inherent variability in the system properties or the external inputs. Global sensitivity analysis (GSA) [20] aims to provide a quantitative assessment of the relative contribution of each uncertainty source to the uncertainty in the model response [21–23]. Much of the GSA literature has focused on variability in the inputs and their effects on output variability; the extension of GSA to include epistemic uncertainty sources (data, model) is recent and sparse [91, 95–98]. Model outputs can have uncertainty even for a fixed input when there exists model uncertainty.

Expanding GSA to consider both aleatory and epistemic uncertainty sources is beneficial in supporting resource allocation decisions. If the contribution of epistemic uncertainty is found to be significant, then it may be valuable to collect more data or refine the physics model to reduce the epistemic uncertainty and thus its contribution to the output uncertainty. Several GSA studies have developed auxiliary variable-based approaches to include both aleatory and epistemic uncertainty sources at a single level instead of using nested simulations, thus achieving both computational efficiency and direct ranking of the different sources of uncertainty to support resource allocation

¹Adapted with permission from: [Kapusuzoglu, B., & Mahadevan, S., “Information fusion and machine learning for sensitivity analysis using physics knowledge and experimental data,” Reliability Engineering & System Safety, vol. 214, no. 107712, 2021.](#)

²© 2021 by Elsevier Ltd.

decision-making. The auxiliary variable is used to transform one-to-many input-output mapping to one-to-one mapping, thus facilitating the computation of Sobol’ indices for both aleatory and epistemic sources [95]. This idea is expanded in [96] to include several epistemic sources, such as input statistical uncertainty, surrogate model error, physics model discrepancy, and numerical solution error, and to systems with time series inputs and outputs.

When the model is computationally expensive, it is often replaced with a surrogate model to facilitate the estimation of Sobol’ indices, since such computation requires many input-output samples from the model; the surrogate model introduces additional uncertainty. Several types of surrogate models are used in the literature, e.g., polynomial chaos expansion (PCE), Gaussian process (GP) regression, neural networks, etc., to train a parametric relationship between the inputs and the outputs. The quality and quantity of the training data affect the accuracy of these surrogate models, which directly affects the uncertainty in the model output [91, 97, 98]. Thus, it is important to also include the contribution of surrogate model uncertainty to the output uncertainty in GSA. In Le Gratiet et al. [91] for example, the Gaussian process surrogate model uncertainty is included in the Sobol’ index estimates using multiple realizations of the GP model prediction, which helps to construct prediction intervals for the Sobol’ index estimates.

Three scenarios of model and data availability can be considered for GSA: (1) use of a physics-based computational model alone, (2) use of available input-output data alone (either from experiments or previous simulations), or (3) use of both physics model and available experimental data. A straightforward model-based approach to estimate Sobol’ indices is to use a double-loop Monte Carlo simulation (MCS) [63]. In order to reduce the cost associated with the double-loop MCS, analytical, spectral and efficient sampling-based methods have been developed. The methodology developed by Sudret [64] approximates the original physics model by a PCE and estimates the Sobol’ indices by using the PCE coefficients. Chen et al. [65] proposed analytical formulas to compute Sobol’ indices using a GP surrogate model with input variables that follow normal or uniform distributions. The improved FAST method [66] combines the classical FAST method [21] with random balanced design [67] for generating samples to evaluate Sobol’ indices.

In some problems, input-output data may be already available instead of having to simulate a physics model expressly for the purpose of GSA. Such data may be available from experiments, or real-world observations, or Markov Chain Monte Carlo (MCMC) sampling during Bayesian model calibration, or MC sampling during reliability analysis, etc. In such cases, data-driven methods have been proposed to directly compute the Sobol’ indices based on available input-output samples instead of simulation runs of the physics model. A GSA method based on ANOVA using factorial design of

experiments is developed by Ginot et al. [99]. The proposed method results in same values as the Sobol' index since the variance decomposition used in the Sobol' index estimations is same as the one used in the classical ANOVA [100]. In high dimensional problems, even the use of a surrogate model for GSA, which repeatedly executes the code by suppressing some variables and running through the range of other variables, may be computationally demanding since the number of executions of the code increases rapidly with the number of inputs [91–94]. The computational cost of most sample-based methods is proportional to the number of model inputs. Li and Mahadevan [101] proposed a modularized method, which has a computational cost that is not proportional to the model input dimension, to estimate the first-order Sobol' indices based on stratification of available input-output samples. DeCarlo et al. [102] proposed an importance sampling approach by introducing weights to different data points to estimate Sobol' indices from available data using Sobol' sequences to reduce the number of simulations; this method computes both first-order and higher order indices, and is able to include correlated inputs. Approximations to the joint probability distribution of inputs and outputs such as multivariate Gaussian, Gaussian copula, and Gaussian mixture have recently been found to give rapid estimation of Sobol' indices [92].

The third scenario is of interest in this chapter, where both a physics-based model and some experimental or real-world data are available. One option, if adequate data is available, is to simply build a regression or machine learning (ML) model based on the observation data, and use this model to perform GSA. Multiple recent studies have pursued data-driven ML models in situations where abundant experimental data or real-world observations are available due to advances in modern sensing techniques. Generally, the construction of data-driven ML models does not require in-depth knowledge of the complex physics inherent in the physical process. ML models can learn complex systems using available observations, but the accuracy of these models depends on the quality and quantity of the data. If the available data is sparse, then the complexity of the process may not be fully captured. Further, since purely data-driven ML models do not explicitly consider physical laws, they can produce physically inconsistent results. In such cases, incorporating physics knowledge within ML models may improve the accuracy and efficiency of GSA computations. The combined use of physics-based and ML models has been shown to achieve more accurate and physically consistent predictions by leveraging the advantages of each method [86, 87, 103, 104].

In this chapter, we incorporate physics knowledge into the ML models to better capture the physics of the process by leveraging physical laws while improving the generalization performance of data-driven models. Two types of strategies are considered for incorporating physics knowledge within ML models: (1) incorporating loss functions in the ML model training to enforce physics

constraints, and (2) pre-training the ML model with data generated by the physics model and then updating it with experimental data. Note that the first strategy does not use the physics model but only constraints for the output to obey physical requirements; whereas, the second strategy explicitly uses the physics computational model. Two types of ML models are considered in this chapter, namely, Gaussian process (GP) and deep neural network (DNN). These two models are selected in order to represent two different kinds of available ML techniques; the GP model is one of the surrogate models commonly used in uncertainty quantification (UQ) studies, and DNN belongs to the emerging class of deep learning algorithms revolutionizing the field of artificial intelligence, spurred by recent advances in sensing, communication and computational resources. Four different physics-informed machine learning (PIML) models are developed for each type (i.e., GP or DNN) to predict the output quantity of interest (QoI), through combinations of the two strategies. The resulting GSA procedure incorporates the effect of uncertainty in the ML or PIML model, and the various models and strategies are compared in terms of accuracy and uncertainty in the GSA results and their computational demand.

In summary, the contributions of this chapter are as follows:

- Physics knowledge and experimental observations are fused in order to maximize the accuracy of sensitivity estimates.
- Two PIML strategies and their combinations are investigated for global sensitivity analysis using both physics knowledge and experimental data.
- Four different models are built for each of GP and DNN, and the uncertainties in these models are included in the Sobol' indices computation.
- The accuracy, uncertainty and computational effort of GSA with different options for the ML and PIML models are evaluated and compared.

4.2 Proposed Methodology

PIML models seek to incorporate physics knowledge or constraints within the data-driven ML models. When a mechanistic, physics-based model is also available, complementary strengths of both mechanistic and ML models can be leveraged in a synergistic manner [103]. In the latter case, the aim is to improve the predictions beyond that of physics-based models or ML models alone by coupling physics-based models with ML models. Thus two different strategies that are developed in Chapter 3 to combine physics knowledge and ML models can be considered: (1) incorporate physics

constraints in the ML models, and (2) pre-train and update the ML models using physics model input-output and experimental data, respectively.

The two proposed strategies to predict the QoI are shown in Fig. 4.1. Figure 4.1(a) shows the first method, where the physical knowledge is included through constraints within the loss function of an ML trained with only experimental data. Figure 4.1(b) shows the second method, where an ML model is first trained with data generated using the physics-based model and then updated using experimental data. Figures 4.1(c) and 4.1(d) show the trained ML model predictions (\hat{Y}) for the two proposed strategies, respectively. The proposed PIML strategies can be applied to any physical system by leveraging the related physical constraints or physics-based models.

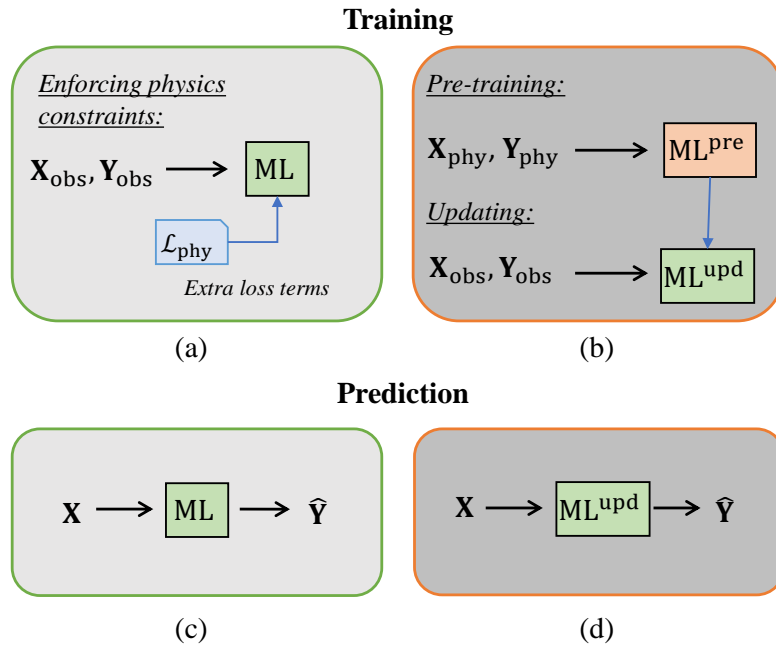


Figure 4.1: PIML strategies: (a) incorporating physics-based loss functions in the ML models to enforce physics constraints, (b) pre-training an ML model with physics model input-output ($\mathbf{X}_{\text{phy}}, \mathbf{Y}_{\text{phy}}$) and updating it with experimental training data ($\mathbf{X}_{\text{obs}}, \mathbf{Y}_{\text{obs}}$), (c) the trained ML model prediction ($\hat{\mathbf{Y}}$), (d) updated ML model prediction ($\hat{\mathbf{Y}}$)

The proposed methodology for sensitivity analysis, using both physics knowledge and experimental data, consists of the following steps:

1. Implementation of PIML strategies in ML models
2. Variance quantification in ML model prediction
3. Sobol' indices computation with ML model prediction variance

The following subsections describe these steps in detail.

4.2.1 Implementation of PIML Strategies in ML Models

Based on the proposed two strategies to incorporate physics knowledge into the ML model, four separate ML models can be constructed for each type of surrogate model considered here (i.e., GP and DNN):

- | | |
|---|--|
| 1. GP | 5. DNN |
| 2. $\text{GP}^{\mathcal{L}_{\text{phy}}}$ | 6. $\text{DNN}^{\mathcal{L}_{\text{phy}}}$ |
| 3. GP^{upd} | 7. DNN^{upd} |
| 4. $\text{GP}^{\text{upd}_1, \mathcal{L}_{\text{phy}}}$ | 8. $\text{DNN}^{\text{upd}, \mathcal{L}_{\text{phy}}}$ |

These different models cover the following options: model trained with experimental data alone, models trained with PIML strategies 1 or 2 alone, and models trained with both PIML strategies together. The implementations of PIML strategies 1, 2, and their combination are different for the GP models vs. the DNN models. Models 5-8 are developed earlier in Chapter 3. Therefore, the following subsection only describe how the PIML strategies can be implemented for GP surrogate models.

4.2.1.1 Implementation of PIML in GP Models

In Model 1, denoted as GP, only experimental observations are used for training. The hyperparameters of the GP model (process variance, correlation length scale along each input dimension, and trend function coefficients, and also measurement error variance if unknown) are optimized during training by maximizing the log marginal likelihood function shown in Eq. (2.13). In calculating the likelihood, the difference between the true response of the system \mathbf{Y}_{true} and the observed response \mathbf{Y}_{obs} is attributed to the observation error $\boldsymbol{\varepsilon}_{\text{obs}}$, which is often treated as a zero-mean Gaussian random variable with variance σ_{obs}^2 .

Model 2, denoted as $\text{GP}^{\mathcal{L}_{\text{phy}}}$, incorporates the first PIML strategy by enforcing physics constraints during the optimization of the GP model hyperparameters. More specifically, the physics constraints are included during the maximization of the log marginal likelihood function (Eq. (2.13)) while inferring the hyperparameters of the GP model. Thus, the training of Model 2 is achieved by maximizing the function in Eq. (4.1):

$$\mathcal{L}_{\text{GP}} = \log p(\mathbf{Y}_{\text{obs}} | \mathbf{X}_{\text{obs}}; \Theta) - \lambda_{\text{phy}} \mathcal{L}_{\text{phy}}(\hat{\mathbf{Y}}), \quad (4.1)$$

where $\hat{\mathbf{Y}}$ is the GP model prediction. Note that since \mathcal{L}_{GP} is to be maximized, the second term corresponding to the physics constraint has a negative sign. Gaussian process modeling under constraints has been studied in the literature [105–109]. Veiga et al. [106] developed a framework that incorporates bound, monotonicity and convexity constraints in GP modeling. Golchi et al. [107] developed a Bayesian approach to GP modeling that incorporates the monotonicity constraint. The need to obtain the monotonicity information at each of the points in the derivative input set can slow down the computation as the input dimension increases since the size of the covariance matrix depends on the input dimension. This chapter does not use the methods described above. Instead, the chapter proposes a different method that penalizes violations of the physics constraints by introducing a regularization term in the likelihood function. This appears to be the first study to apply the penalty approach to the likelihood function of the GP model. Further, the computational effort of the proposed method (i.e., the calculation of the regularization term) does not increase with the problem size since the regularization term does not need the inverse of the covariance matrix.

The current work considers two different approaches for the second PIML strategy. Both approaches use a pre-trained model, obtained using the physics model input-output data. In the first approach, the model parameters are updated using the experimental data; and in the second approach, a discrepancy correction term is added to the pre-trained model. The first step of pre-training using data generated by the physics model can be thought of as similar to a lower fidelity model, and the second step of improving the pre-trained model (either by parameter updating or by adding a discrepancy term) can be thought of as similar to incorporating higher fidelity data (experimental data, in this case) to improve the model. Various multi-fidelity modeling strategies with different combinations of the low- and high-fidelity models have been studied in the literature, such as filtering, fusion, and adaptation [110]. The discrepancy correction approach pursued here adopts the simplest strategy, namely, additive correction, where a model discrepancy term is added to the low-fidelity model [111]:

$$f_{\text{HF}}(\mathbf{X}) = f_{\text{LF}}(\mathbf{X}) + \delta(\mathbf{X}; \boldsymbol{\theta}_{\delta}) \quad (4.2)$$

where $f_{\text{HF}}(\cdot)$ and $f_{\text{LF}}(\cdot)$ are the high and low fidelity models, respectively, \mathbf{X} is the input to the model, and $\boldsymbol{\theta}_{\delta}$ are the parameters of the discrepancy correction term. The correction term can be obtained using any suitable surrogate modeling technique.

Model 3, denoted as GP^{upd} , pursues the first approach of the second PIML strategy, where a GP model is pre-trained using the coupled multi-physics model input-output and then updated with experimental data. Then, the model parameters of this pre-trained network are updated using the

experimental data.

An alternative approach, denoted as GP^{MF} , pursues the second approach of the second PIML strategy, i.e., it pre-trains a GP surrogate model with data generated from the physics model, then improves the surrogate using experimental data. When the physics model is computationally expensive, it is replaced by a cheaper surrogate model. In Model 3, a GP surrogate model is used to approximate the original physics model. The accuracy of the surrogate model prediction depends on the quality and quantity of the training data generated by the original physics model. Following the procedure described in Section 2.3, the surrogate model error ($\epsilon_\delta(\mathbf{X})$) can be incorporated as follows:

$$\mathbf{Y}_m(\mathbf{X}) = \hat{\mathbf{Y}}_m(\mathbf{X}) + \epsilon_\delta(\mathbf{X}), \quad (4.3)$$

where $\hat{\mathbf{Y}}_m$ is the surrogate model prediction.

A simple situation is considered in this section. There is no calibration of the physics model parameters here; only the discrepancy term is needed. (In other words, the physics model parameters are already established). In that case, the model discrepancy can be evaluated for different input values of experimental tests and realizations of observation errors as follows:

$$\delta(\mathbf{X}) = \mathbf{Y}_{\text{obs}}(\mathbf{X}) - \epsilon_{\text{obs}}(\mathbf{X}) - \hat{\mathbf{Y}}_m(\mathbf{X}) - \epsilon_\delta(\mathbf{X}). \quad (4.4)$$

Moving the surrogate model error $\epsilon_\delta(\mathbf{X})$ to the left-hand side, we can express the difference between the actual response and GP model prediction as

$$\hat{\delta}(\mathbf{X}) = \delta(\mathbf{X}) + \epsilon_\delta(\mathbf{X}) = \mathbf{Y}_{\text{obs}}(\mathbf{X}) - \epsilon_{\text{obs}}(\mathbf{X}) - \hat{\mathbf{Y}}_m(\mathbf{X}). \quad (4.5)$$

In this work, a second GP model is trained for $\hat{\delta}(\mathbf{X})$ in terms of the inputs. Thus two GP models are trained in Model 3. The first GP model is constructed using the physics model input-output data, and predicts $\hat{\mathbf{Y}}_m$. The second GP model is constructed using the experimental data and the corresponding surrogate model predictions, and predicts $\hat{\delta}$ (the difference between the surrogate model prediction and actual system response).

The GP model for the model discrepancy captures the combined contribution of measurement error, physics and surrogate model errors for a given prediction. Thus, the predictions of the first GP model (pre-trained) are corrected with the second GP model predictions ($\hat{\delta}$) representing the

model discrepancy term and can be written as

$$\hat{\mathbf{Y}}(\mathbf{X}) = \hat{\mathbf{Y}}_m(\mathbf{X}) + \hat{\delta}. \quad (4.6)$$

Model 4, denoted as $\text{GP}^{\text{upd}, \mathcal{L}_{\text{phy}}}$, combines both PIML strategies for GP, where the optimized model parameters of the pre-trained model based on the physics model input-output are used as the initial values. The pre-trained model is corrected with $\hat{\delta}$ and the physics constraints are enforced within the correction. The first GP model is trained using the input-output samples from the physics model to predict $\hat{\mathbf{Y}}_m$. Then, using the experimental data, the second GP model is constructed by enforcing the physics constraints during the optimization of its hyperparameters. These model parameters are updated using the experimental data by minimizing the augmented loss function shown in Eq. (4.1) which consists of both the training loss function and the physics constraint loss terms. The prediction of the updated model is $\hat{\mathbf{Y}}$, given by the sum of the predictions of the two GP models.

4.2.2 Variance of GP and DNN Prediction

In general, every surrogate model has uncertainty in prediction, whether acknowledged or not. In the GP models, the prediction at a given input is expressed by a normal distribution with a mean and variance. In order to quantify the uncertainty in the GP prediction, we can sample multiple realizations of the Gaussian process. Note that this only captures the variance of the GP prediction, not the bias, which can be evaluated by comparing against validation data.

In the DNN models, the estimates of the model parameters (neuron weights \mathbf{w}) have uncertainty, and this uncertainty depends on the available training data. When the neural network parameters are represented using distributions (to reflect the epistemic uncertainty) instead of deterministic values, the model is referred to as a Bayesian neural network (BNN) [40–42] (see Section 2.2.3).

The sensitivity estimate results depend on the dropout rate. The main reason for this is that the model is regularized and it underfits the data as it is over-regularized. Further, both the accuracy and uncertainty of the sensitivity estimates also depend on the number of training epochs, and the architecture of the network. If the model is not fully trained, it will also result in underfitting, leading to larger bias and variance in the prediction.

4.2.3 Sobol' Indices Computation with Model Uncertainty

This section discusses the incorporation of ML model prediction variance within the estimation of Sobol' indices using the GP and DNN models.

When the training data is noise-free, the GP predictions at the training points have zero variance and at other points the variance is non-zero. The prediction at any point is given by a normal distribution with a mean and variance. This prediction uncertainty can be captured by sampling multiple realizations of the GP model, which can then be used in GSA. The model uncertainty pertaining to the GP model is propagated to the Sobol' index calculations using the following estimator (see [91]):

$$S_m^{\text{GP}} = \frac{V_{X_i}(E_{\mathbf{X}}[\mathbf{y}_P(\mathbf{X})|\mathbf{X}'])}{V(\mathbf{y}_P(\mathbf{X}))} \approx \frac{\frac{1}{m} \sum_{k=1}^m \mathbf{y}_P(\mathbf{X}_k) \mathbf{y}_P(\mathbf{X}'_k) - \frac{1}{m} \sum_{k=1}^m \mathbf{y}_P(\mathbf{X}_k) \frac{1}{m} \sum_{k=1}^m \mathbf{y}_P(\mathbf{X}'_k)}{\frac{1}{m} \sum_{k=1}^m \mathbf{y}_P(\mathbf{X}_k)^2 - [\frac{1}{m} \sum_{k=1}^m \mathbf{y}_P(\mathbf{X}_k)]^2}, \quad (4.7)$$

where $\mathbf{y}_P(\mathbf{X})$ is a realization of the predictive distribution shown in Eq. (2.11) trained using experimental data, and \mathbf{X}_k and \mathbf{X}'_k are the k th samples of the random vectors \mathbf{X} and \mathbf{X}' .

The distribution of S_m^{GP} can be computed by sampling N_Z realizations from the Gaussian predictive distribution $\mathbf{Y}_P(\mathbf{X})$ numerically using Algorithm 1.

Algorithm 1 Estimation of the distribution of S_m^{GP} using GP models

- 1: Generate two samples \mathbf{X}_k and \mathbf{X}'_k ($k = 1, \dots, m$) of the random vectors \mathbf{X} and \mathbf{X}' .
 - 2: for $p = 1, 2, \dots, N_Z$ do
 - 3: Sample a realization $\mathbf{y}_P(\mathbf{x})$ of $\mathbf{Y}_P(\mathbf{X})$ with $\mathbf{x} = \{(x_k)_{k=1, \dots, m}, (x'_k)_{k=1, \dots, m}\}$.
 - 4: Compute $\hat{S}_{m,p}^{\text{GP}}$ using Eq. (4.7).
 - 5: end for
- return $(\hat{S}_{m,p}^{\text{GP}})_{p=1,2, \dots, N_Z}$.
-

The output of Algorithm 1 $(\hat{S}_{m,p}^{\text{GP}})_{p=1,2, \dots, N_Z}$ is a sample of size N_Z , where m is the number of Monte Carlo samples. Thus, the mean and variance of sensitivity estimates obtained using the GP models are defined as follows, respectively:

$$\begin{aligned} \mu_{S_m^{\text{GP}}} &= \frac{1}{N_Z} \sum_{p=1}^{N_Z} \hat{S}_{m,p}^{\text{GP}}, \\ \sigma_{S_m^{\text{GP}}}^2 &= \frac{1}{N_Z} \sum_{p=1}^{N_Z} (\hat{S}_{m,p}^{\text{GP}} - \mu_{S_m^{\text{GP}}})^2. \end{aligned} \quad (4.8)$$

A similar approach can be implemented in the DNN models with the use of MC dropout (with a

chosen dropout rate). However, in contrast to the GP models, where we sample from a multivariate normal distribution to quantify the uncertainty in the Sobol' index estimates, the sampling implementation is different in the DNN models. In the DNN models, we randomly set units of the network to zero and generate predictions using the remaining units of the network as shown in Algorithm 2. The neuron weights can be drawn from the approximate posterior $\hat{\mathbf{w}} \sim q_{\theta}(\mathbf{w})$ to obtain the model outputs of a DNN with MC dropout denoted as $\mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}) = \hat{\mathbf{Y}}$, where $\hat{\mathbf{Y}}$ is the predictive mean,

The predictive posterior given in Eq. 2.15 can be defined as follows:

$$p(\hat{\mathbf{Y}}|\hat{\mathbf{X}}, \mathbf{X}_T, \mathbf{Y}_T) = \mathcal{N}(\hat{\mathbf{Y}}; \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}), \sigma^2 \mathbf{I}) \quad (4.9)$$

where $\sigma^2 = (2N\lambda)/((1-p_d)l^2)$ is the noise term [47], l being the prior length-scale, and λ is the regularization strength used in typical loss functions. Dropout can be interpreted as a variational Bayesian approximation and the minimization objective is defined by [112]

$$\mathcal{L}(\theta, p_d) = -\frac{1}{N} \sum_{i=1}^N \log p(\hat{\mathbf{Y}}_i | \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}_i)) + \frac{1-p_d}{2N} \|\theta\|^2 \quad (4.10)$$

where θ is the set of distribution's parameters to be optimized (i.e., weights of the network).

The predictive mean and predictive uncertainty are estimated by collecting the results of stochastic forward passes through the model. The mean prediction of the model with N_d samples can be approximated by

$$\mathbb{E}(\mathbf{Y}) \approx \frac{1}{N_d} \sum_{t=1}^{N_d} \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}), \quad (4.11)$$

and the variance of the prediction is estimated by

$$\text{Var}(\mathbf{Y}) \approx \sigma^2 + \frac{1}{N_d} \sum_{t=1}^{N_d} \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X})^T \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}) - \mathbb{E}(\mathbf{Y})^T \mathbb{E}(\mathbf{Y}). \quad (4.12)$$

Similar to Eq. (4.7), the uncertainty in the DNN model can be propagated to the sensitivity calculations using the following estimator:

$$S_m^{\text{DNN}} = \frac{\frac{1}{m} \sum_{k=1}^m \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}_k) \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}'_k) - \frac{1}{m} \sum_{k=1}^m \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}_k) \frac{1}{m} \sum_{k=1}^m \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}'_k)}{\frac{1}{m} \sum_{k=1}^m \mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}_k)^2 - \frac{1}{m} \sum_{k=1}^m (\mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X}_k))^2}, \quad (4.13)$$

where $\mathcal{G}^{\hat{\mathbf{w}}}(\mathbf{X})$ denotes the deep neural network (DNN) with MC dropout and all the other terms have the same definition as Eq. (4.7). The model uncertainty is propagated to the calculation of the sensitivity estimates by directly using the results based on stochastic forward passes through a

dropout-reduced DNN instead of the predictive mean and predictive uncertainty.

Algorithm 2 Estimation of the distribution of S_m^{DNN} using DNN models with MC dropout.

- 1: Generate samples \mathbf{X}_k and \mathbf{X}'_k ($k = 1, \dots, m$) of the random vectors \mathbf{X} and \mathbf{X}' .
 - 2: for $p = 1, 2, \dots, N_d$ do
 - 3: Perform a stochastic forward pass through the network $\mathcal{G}^{\hat{w}}(\mathbf{X})$ using MC dropout and calculate the model prediction $\hat{\mathbf{Y}} = \mathcal{G}^{\hat{w}}(\mathbf{X})$.
 - 4: Compute $\hat{S}_{m,p}^{\text{DNN}}$ using Eq. (4.13).
 - 5: end for
- return $(\hat{S}_{m,p}^{\text{DNN}})_{p=1,2,\dots,N_d}$.
-

The output of Algorithm 2 $(\hat{S}_{m,p}^{\text{DNN}})_{p=1,2,\dots,N_d}$ is a sample of size N_d . Thus, the mean and variance of sensitivity estimates obtained using DNN models are defined as follows, respectively:

$$\begin{aligned} \mu_{S_m^{\text{DNN}}} &= \frac{1}{N_d} \sum_{p=1}^{N_d} \hat{S}_{m,p}^{\text{DNN}}, \\ \sigma_{S_m^{\text{DNN}}}^2 &= \frac{1}{N_d} \sum_{p=1}^{N_d} (\hat{S}_{m,p}^{\text{DNN}} - \mu_{S_m^{\text{DNN}}})^2. \end{aligned} \quad (4.14)$$

In summary, two types of PIML strategies (developed in Chapter 2) are implemented in two types of ML models (GP and DNN), in order to evaluate the accuracy and uncertainty of the sensitivity estimates in GSA. Eight different PIML models are developed by leveraging the two PIML strategies. The accuracy of these models can be assessed by comparison against validation data, whereas the variance of the sensitivity estimates can be quantified using Algorithms 1 and 2 and Eqs. 4.7 to 4.14. The different models have different training strategies and different numbers of parameters, both of which will affect the accuracy and uncertainty of the sensitivity estimates. These differences are assessed in detail in the next section.

4.3 Numerical Example

4.3.1 Training Details of the ML Models

The output QoI is the porosity of the printed part, and the inputs are two process parameters, namely nozzle temperature and speed. The problem setup is described in Section 3.3.1. Models of two physical phenomena (heat transfer and sintering models that are defined in Section 2.5) are combined to predict the porosity given the values of two process parameters: nozzle temperature and nozzle speed. Thus, porosity values are obtained using both experiments and physics models.

The basic ML models, namely Model 1 (for GP) and Model 5 (for DNN) are simply trained with the 39 sets of process inputs (temperature and speed) and output (porosity). In the training of Model 2 (GP ^{\mathcal{L}_{phy}}) and Model 6 (DNN ^{\mathcal{L}_{phy}}), we impose two physics constraints (i.e., two separate loss function terms, $\mathcal{L}_{\text{phy},k}(\hat{\mathbf{Y}})$, where $k = \{1, 2\}$ and $\hat{\mathbf{Y}}$ is the porosity prediction). The corresponding loss function terms are defined as

$$\begin{aligned}\mathcal{L}_{\text{phy},1}(\hat{\mathbf{Y}}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(-\hat{Y}_i), \\ \mathcal{L}_{\text{phy},2}(\hat{\mathbf{Y}}) &= \frac{1}{N} \sum_{i=1}^N \text{ReLU}(\hat{Y}_i - \phi_{0,i}),\end{aligned}\tag{4.15}$$

considering physics violations related to the porosity in all the N samples. In the first loss function, a negative value of porosity is treated as a physics violation. The second loss function penalizes the model when the predicted final porosity \hat{Y}_i is greater than the initial porosity $\phi_{0,i}$ of the i th part. This is based on the physics knowledge that the total void area decreases as the bond formation takes place. Thus, the porosity predictions are ensured to be physically meaningful with the inclusion of these physics-based penalty terms.

The overall “loss” function of the GP model is

$$\mathcal{L}_{\text{GP}} = \mathcal{L}_{\text{GP}} - \lambda_{\text{phy},1}^{\text{GP}} \mathcal{L}_{\text{phy},1}(\hat{\mathbf{Y}}) - \lambda_{\text{phy},2}^{\text{GP}} \mathcal{L}_{\text{phy},2}(\hat{\mathbf{Y}}).\tag{4.16}$$

Note that the GP model parameters are obtained by maximizing the above function.

The overall loss function of the DNN model is

$$\mathcal{L}_{\text{DNN}} = \mathcal{L}_{\text{DNN}} + \lambda_{\text{phy},1}^{\text{DNN}} \mathcal{L}_{\text{phy},1}(\hat{\mathbf{Y}}) + \lambda_{\text{phy},2}^{\text{DNN}} \mathcal{L}_{\text{phy},2}(\hat{\mathbf{Y}}).\tag{4.17}$$

Note that the DNN model parameters are obtained by minimizing the above function.

In Model 3 (GP^{upd}) and Model 7 (DNN^{upd}), the ML models are pre-trained using the multi-physics model input-output. The pre-trained ML models are then updated using the experimental data. The training data for pre-training consists of 1310 input parameter combinations over a range of experimental values, i.e., ($210^\circ\text{C} \leq T \leq 260^\circ\text{C}$, $15 \text{ mm/s} \leq S \leq 46 \text{ mm/s}$). Note that there are only 39 physical experiments available; this is one of the advantages of the pre-training/updated strategy, where the pre-training can be over a much larger set of input combinations, thus improving the generalization performance of the updated model. The input data are normalized prior to the training of the ML models (i.e., the output quantity porosity is dimensionless and between 0 and 1).

Model 4 ($\text{GP}^{\text{upd}, \mathcal{L}_{\text{phy}}}$) combines both PIML strategies for GP, and consists of two GP models: (i) the first GP model is trained using the physics model input-output samples consisting of 1310 input parameter combinations; and (ii) the second GP model is built for the discrepancy between the first GP model prediction and the actual system response using the experimental data, by maximizing the function shown in Eq. (4.16) to optimize the hyperparameters of the second GP model.

Model 8 $\text{DNN}^{\text{upd}, \mathcal{L}_{\text{phy}}}$, which is a combination of the two PIML strategies, uses the DNN model parameters trained using the physics model input-output as the initial values. Then, during the updating phase with the experimental data, these parameters are updated by minimizing the loss function shown in Eq. (4.17).

The four GP models (Models 1 to 4) were implemented using Python. The optimization of the hyperparameters were performed using the scikit-optimize package. The multipliers of the physics constraint terms of models 2 ($\text{GP}^{\mathcal{L}_{\text{phy}}}$) and 4 ($\text{GP}^{\text{upd}, \mathcal{L}_{\text{phy}}}$) were chosen as $(\lambda_{\text{phy},1}^{\text{GP}}, \lambda_{\text{phy},2}^{\text{GP}}) = (50, 50)$ based on a cross-validation test. The Automatic Relevance Determination (ARD) squared exponential function [37] was used as the covariance function for all the GP models.

The four DNN models (Models 5 to 8) were implemented using the Keras package [90] with Tensorflow in the backend. The hyperparameters of each model were tuned with grid search and the multipliers of the physics constraint terms in Model 6 ($\text{DNN}^{\mathcal{L}_{\text{phy}}}$) and Model 8 ($\text{DNN}^{\text{upd}, \mathcal{L}_{\text{phy}}}$) were chosen as $(\lambda_{\text{phy},1}^{\text{DNN}}, \lambda_{\text{phy},2}^{\text{DNN}}) = (0.01, 0.01)$ based on a cross-validation test. Fully-connected DNN models with 2 hidden layers and 5 neurons in each hidden layer were constructed. The Rectified Linear Unit (ReLU) activation function and Adam optimizer were used to perform stochastic gradient descent for 300 epochs in learning the model parameters. The dropout rate for the DNN models was chosen to be 0.05, for reasons as explained below.

4.3.2 Comparison of Computational Effort

The computational costs of different models for training and estimation of Sobol' indices based on 5000 MC samples with a fixed number of experimental data ($n = 39$) is given in Table. 4.1. Among the GP models, the time it takes for training as well as computation of Sobol' indices using Models 2-4 is significantly greater than Model 1. The reason for the difference between the training time of GP and GP^{upd} is the pre-training phase, where a large amount of physics input-output samples used. Whereas, the difference between the training time of GP and $\text{GP}^{\mathcal{L}_{\text{phy}}}$ is due to the inclusion of physics constraints, which makes it harder for the optimization to find optimal hyperparameters. Interestingly, the training time for the DNN models ranges from 20 to 55 sec on the same desktop computer as used for the GP model (Intel® Xeon® CPU E5-1660 v4@3.20GHz with 32 GB RAM and

GPU NVIDIA Quadro K620 with 2 GB). Among the DNN models, the reasons for the increased training time of models 6-8 compared to Model 5 are the same as for the GP models. The Sobol' index estimations based on 5000 samples take approximately 1-2 minutes for the DNN models (models 5-8), whereas the GP predictions take much longer because the covariance matrix needs to be stored and inverted.

Table 4.1: Computational effort of eight models for training and estimation of first-order and total effect Sobol' indices using 5000 MC samples with $n = 39$ number of observations

Models	Training [in minutes]	Sobol' indices calculation [in minutes]
1. GP	1	3
2. GP $^{\mathcal{L}_{\text{phy}}}$	2	5
3. GP $^{\text{upd}}$	3	7
4. GP $^{\text{upd}, \mathcal{L}_{\text{phy}}}$	3	8
5. DNN	.3	1
6. DNN $^{\mathcal{L}_{\text{phy}}}$.7	2
7. DNN $^{\text{upd}}$.5	2
8. DNN $^{\text{upd}, \mathcal{L}_{\text{phy}}}$	1	2

4.3.3 Comparison of Accuracy

In order to compare the accuracy of the eight different models, the models are trained with different amounts of experimental observations ($n = (5, 10, 15, 20, 30)$), and the remaining 9 observations are used to compute the errors; the root mean square error (RMSE) based on the 9 validation samples is reported as the accuracy measure for comparison. The mean and one standard deviation RMSE values for the four GP models and the four DNN models are shown in Tables 4.2 and 4.3 respectively, for different values of n . To further validate the accuracy of the models, the data set is divided into two subsets for cross-validation; k -fold cross-validation is performed by splitting the data set into sets for model training and cross-validation, and these sets are selected randomly $k = 10$ different times. The average cross-validation accuracy of the models over the 10 folds (random shuffles) is assessed by evaluating the average RMSE (see Table 4.4).

The results in Tables 4.2, 4.3 and 4.4 show that the use of PIML strategies in DNN models improves the performance, and the improvement is relatively larger as the amount of observed data n gets smaller. For $n=20$ and $n=30$, the basic DNN model is as accurate as the DNN models with the PIML strategies; whereas for smaller values of n , the DNN models with the PIML strategies are significantly more accurate. This clearly indicates the benefit of PIML over basic ML for the DNN models. However, the GP models incorporating PIML strategies do not show significant improvement. This is because the GP models have a much smaller number of parameters compared

Table 4.2: Effect of different amounts of training data on the RMSE of the GP models

Model	$n = 5$	$n = 10$	$n = 15$	$n = 20$	$n = 30$
1. GP	0.020(± 0.004)	0.026(± 0.005)	0.021(± 0.001)	0.021(± 0.002)	0.020(± 0.001)
2. GP $^{\mathcal{L}_{\text{phy}}}$	0.019(± 0.004)	0.026(± 0.004)	0.024(± 0.004)	0.021(± 0.002)	0.019(± 0.001)
3. GP $^{\text{upd}}$	0.021(± 0.003)	0.028(± 0.003)	0.020(± 0.001)	0.020(± 0.001)	0.019(± 0.001)
4. GP $^{\text{upd}, \mathcal{L}_{\text{phy}}}$	0.021(± 0.005)	0.023(± 0.003)	0.022(± 0.003)	0.020(± 0.002)	0.019(± 0.001)

Table 4.3: Effect of different amounts of training data on the RMSE of the DNN models

Model	$n = 5$	$n = 10$	$n = 15$	$n = 20$	$n = 30$
5. DNN	0.027(± 0.007)	0.017(± 0.009)	0.019(± 0.006)	0.013(± 0.003)	0.013(± 0.003)
6. DNN $^{\mathcal{L}_{\text{phy}}}$	0.017(± 0.007)	0.015(± 0.008)	0.014(± 0.005)	0.014(± 0.003)	0.013(± 0.002)
7. DNN $^{\text{upd}}$	0.014(± 0.002)	0.009(± 0.002)	0.014(± 0.001)	0.013(± 0.001)	0.013(± 0.001)
8. DNN $^{\text{upd}, \mathcal{L}_{\text{phy}}}$	0.014(± 0.002)	0.009(± 0.001)	0.014(± 0.001)	0.014(± 0.001)	0.013(± 0.001)

to the DNN models, and the achievable accuracy of any ML model is constrained by the number of model parameters (in addition to model form).

In addition to the increased number of parameters, the DNN models have additional advantages in terms of training epochs and dropout rate that affect the prediction accuracy. As mentioned earlier, the optimum number of training epochs was found to be 300. Regarding dropout rate, the RMSE values (based on 9 validation samples) of DNN models with MC dropout for different dropout rates are shown in Fig. 4.2. The lowest RMSE values for all four models are obtained between 0.005-0.05. On the other hand, the Sobol' index estimates were found to be similar within this range of dropout rate, for each of the four DNN models. Therefore, a dropout rate of 0.05 was chosen for the reporting of GSA results, since a higher dropout rate results in smaller sub-networks, therefore a smaller number of parameters and faster training. By comparing Tables 4.2 and 4.3, it is seen that at the dropout rate of 0.05, the DNN models are more accurate compared to the GP models.

The Sobol' indices estimates obtained using the two approaches of the second PIML strategy are compared in Figs. 4.3 and 4.4. The results show that both approaches converge to similar first-order and total effect sensitivity index estimates. Thus, in the rest of the chapter the first approach of the second PIML strategy is used for both GP and DNN models, i.e., GP $^{\text{upd}}$ and DNN $^{\text{upd}}$.

Table 4.4: Tenfold cross-validation average RMSE results of GP and DNN models

	Models							
	1. GP	2. GP $^{\mathcal{L}_{\text{phy}}}$	3. GP $^{\text{upd}}$	4. GP $^{\text{upd}, \mathcal{L}_{\text{phy}}}$	5. DNN	6. DNN $^{\mathcal{L}_{\text{phy}}}$	7. DNN $^{\text{upd}}$	8. DNN $^{\text{upd}, \mathcal{L}_{\text{phy}}}$
$n = 39$	0.021(± 0.002)	0.020(± 0.002)	0.020(± 0.001)	0.019(± 0.002)	0.015(± 0.004)	0.013(± 0.004)	0.013(± 0.003)	0.013(± 0.003)

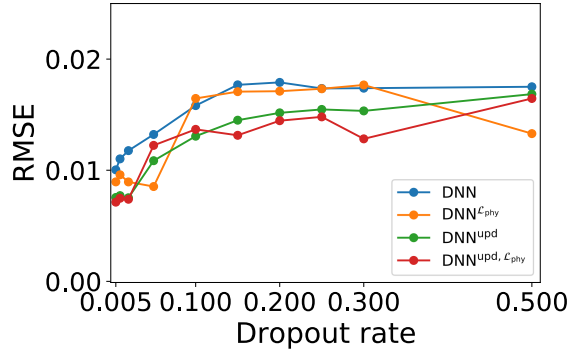


Figure 4.2: RMSE values of DNN models with varying dropout rates

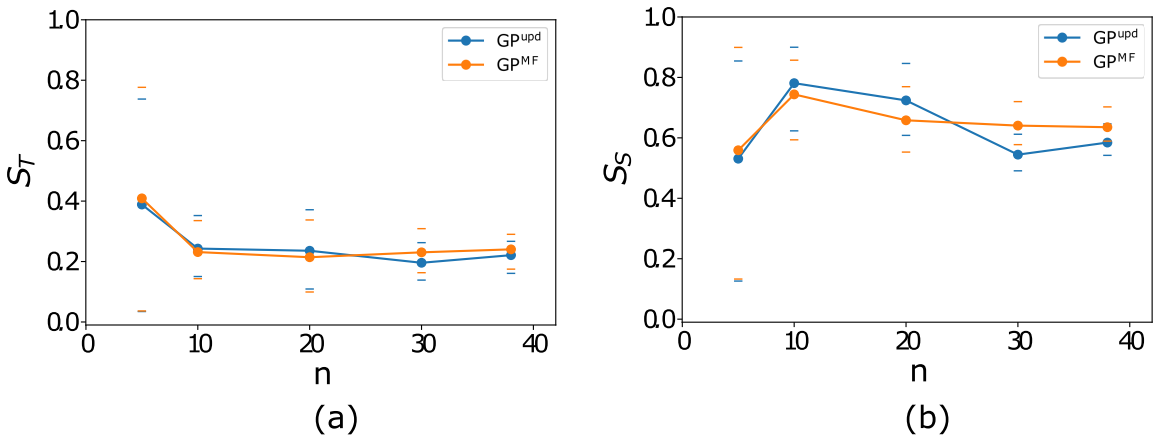


Figure 4.3: First-order sensitivity index estimators for (a) nozzle temperature, and (b) nozzle speed, using two different pre-training and updating approaches for GP

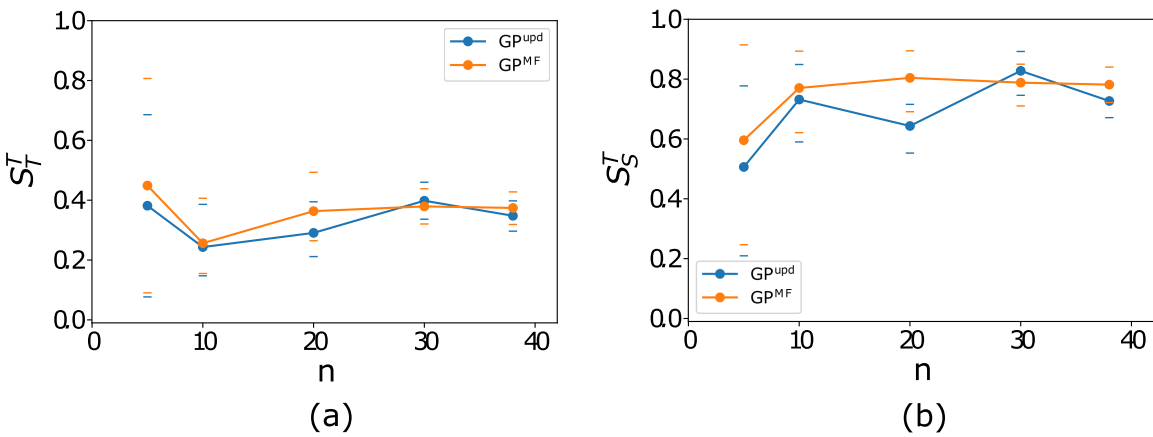


Figure 4.4: Total effect sensitivity index estimates for (a) nozzle temperature, and (b) nozzle speed, using two different pre-training and updating approaches for GP

4.3.4 GSA Results using GP Models

The Sobol' index computations with the GP models (1-4) are based on 5000 MC samples and 100 realizations of the Gaussian process. The effect of the number of experimental observations (used to train the GP models) on the first-order Sobol' index estimates from the four GP models is illustrated in Fig. 4.5. And the total effect Sobol' index estimates from the GP models for different numbers of experimental training data are shown in Fig. 4.6. The mean values of sensitivity estimates based on the GP model predictions are denoted with solid dots at a given number of observations n . The sensitivity results are reported for two input variables, nozzle temperature and nozzle speed; the output quantity of interest is porosity.

The 95% prediction intervals are represented with bars above and below the solid dots for the corresponding model. The bounds in the sensitivity estimates are calculated using the mean predictions based on the 100 realizations of the GP models. As expected, the prediction intervals decrease as increasing amounts of experimental data are used to train the GP models. Further, all four models converge to similar first-order and total effect sensitivity estimates for both printer nozzle temperature and speed. The relative individual contribution (at $n=39$) of nozzle speed to the variance of the porosity (≈ 0.65) is greater than that of the nozzle temperature (≈ 0.25) and their sum is ≈ 0.9 . And the sum of their total effect indices (which capture parameter interactions) is slightly above 1.0, indicating that the interaction effect is small.

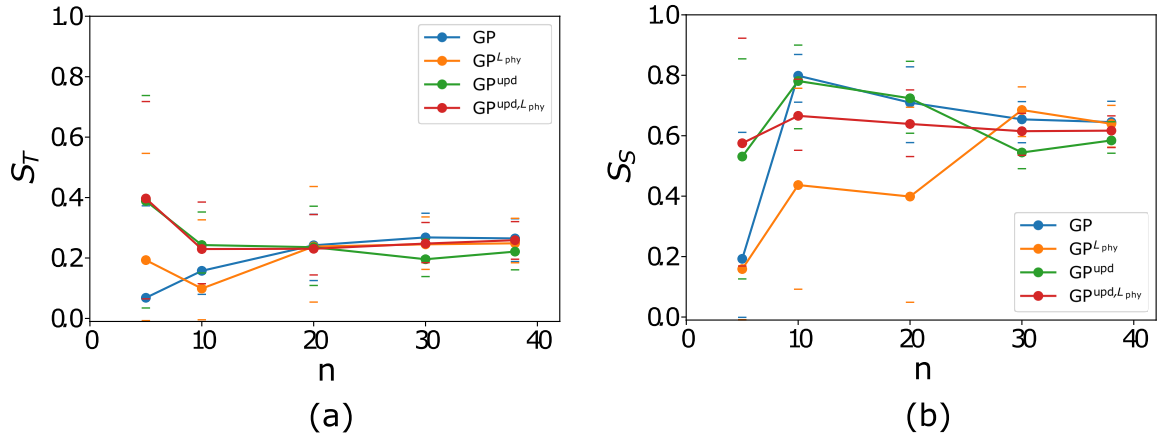


Figure 4.5: First-order sensitivity index estimates for (a) nozzle temperature, and (b) nozzle speed, using the GP models

For further clarity regarding uncertainty, numerical values of the prediction bounds of first-order sensitivity estimates of nozzle temperature obtained using 100 realizations of the GP models are shown in Table 4.5. The results indicate that prediction intervals obtained using the PIML models

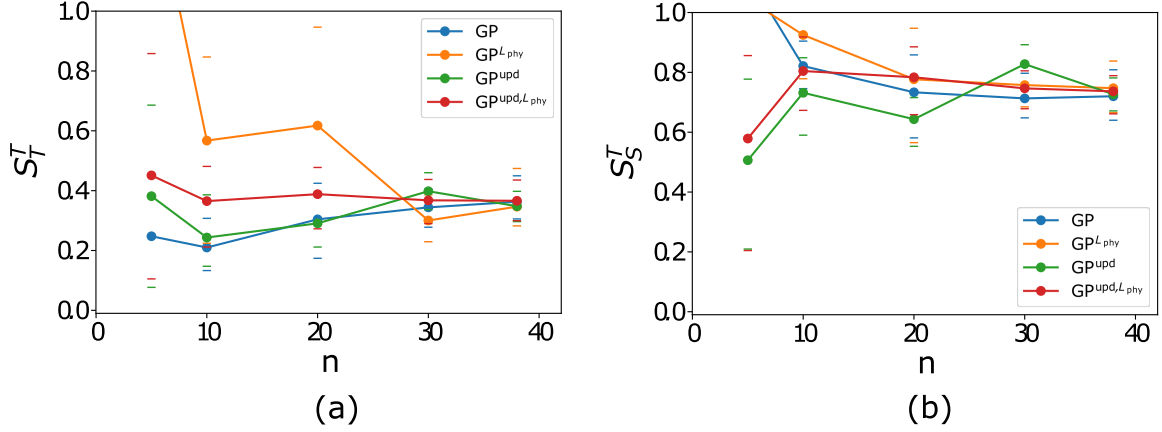


Figure 4.6: Total effect sensitivity index estimates for (a) nozzle temperature, and (b) nozzle speed, using the GP models

3 and 4 (GP^{upd} and $\text{GP}^{\text{upd},\mathcal{L}_{\text{phy}}}$) converge to the bounds obtained using 39 number of observations for training the models faster than the first two models. Note that the upper 95% bound for Model 2 ($\text{GP}^{\mathcal{L}_{\text{phy}}}$) converges to its final value (0.33) within 10 experimental observations.

Table 4.5: First-order sensitivity estimate prediction bounds of nozzle temperature for different amounts of experimental training data, using the GP models

Models	Lower 95% confidence limit					Upper 95% confidence limit				
	n=5	n=10	n=20	n=30	n=39	n=5	n=10	n=20	n=30	n=39
1. GP	0.00	0.08	0.13	0.20	0.19	0.37	0.25	0.34	0.35	0.33
2. $\text{GP}^{\mathcal{L}_{\text{phy}}}$	0.00	0.00	0.05	0.16	0.18	0.55	0.33	0.43	0.34	0.33
3. GP^{upd}	0.03	0.14	0.10	0.16	0.17	0.77	0.33	0.34	0.31	0.29
4. $\text{GP}^{\text{upd},\mathcal{L}_{\text{phy}}}$	0.06	0.11	0.14	0.18	0.20	0.71	0.38	0.34	0.31	0.32

4.3.5 GSA Results using DNN Models with MC Dropout

The Sobol' index computations with the DNN models (5-8) are based on 5000 MC samples and 100 stochastic forward passes through the networks. The calculated first-order sensitivity estimates of temperature and speed, S_T and S_S respectively, for different numbers of experimental observations (training data) $n = (5, 10, 20, 30, 39)$ are shown in Fig. 4.7. The distributions of sensitivity estimates are obtained using MC dropout predictions based on 100 stochastic forward passes through the networks for different number of observations. The mean values of sensitivity estimates are represented with solid dots and the 95% bounds are denoted with bars above and below the solid dots for the corresponding model. Similarly, the calculated total effect sensitivity estimates S_T and S_S for different values of n are illustrated in Fig. 4.8. Similar to the GP results, the difference between the total effect and first-order indices of process inputs is negligible, which indicates that their interaction is

not significant.

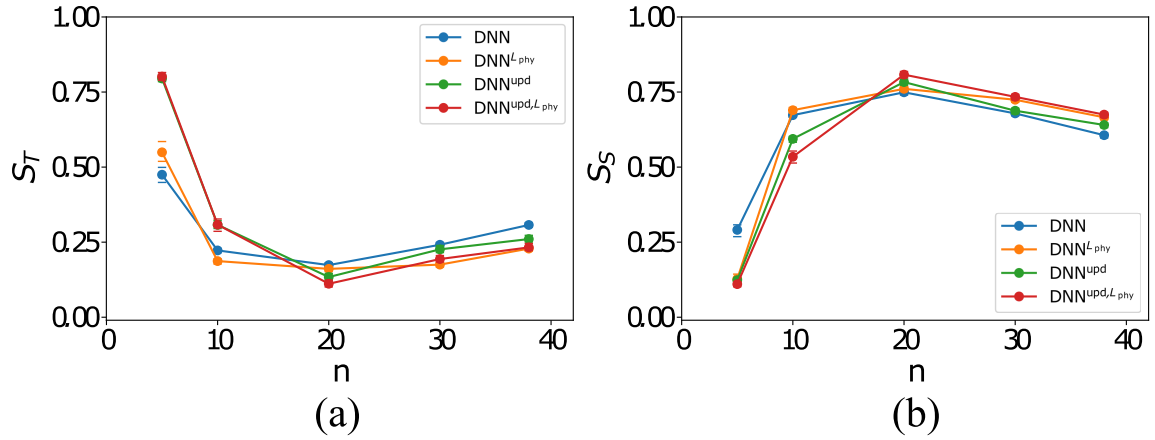


Figure 4.7: First-order sensitivity index estimators for (a) nozzle temperature, and (b) nozzle speed, using DNN models with MC dropout

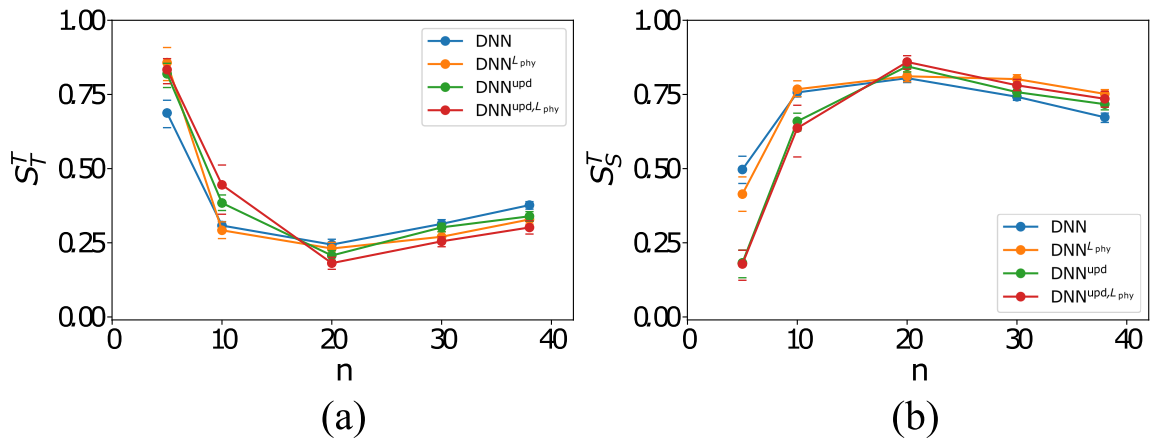


Figure 4.8: Total effect sensitivity index estimators for (a) nozzle temperature, and (b) nozzle speed, using DNN models with MC dropout

All DNN models converge to similar first-order and total effect sensitivity estimates for both inputs, and these values are consistent with the results obtained using GP models. For this problem with two inputs and one output, 39 experimental observations appear adequate to train even the basic DNN model to achieve similar performance as the other physics-informed models; in fact, the results are similar even at 20 observations. However, the superior accuracy of the physics-informed models becomes apparent if only a small number of experiments are available, say $n = 5$ or $n = 10$, as shown in Table 4.3 and discussed earlier in Section 4.3.3.

For further clarity regarding uncertainty, numerical values of the prediction bounds of first-order sensitivity estimates of nozzle temperature obtained using 100 forward passes through the DNN models are given in Table 4.6. The results show that prediction intervals obtained using all models

show a similar trend. The 95% bounds for all models are significantly narrower than the ones obtained using the GP models. The uncertainty in the sensitivity estimates due to the DNN models is almost negligible when more than 10 number of observations are used to train the models.

Table 4.6: First-order sensitivity estimate prediction bounds of nozzle temperature for different amounts of experimental training data, using the DNN models

Models	Lower 95% confidence limit					Upper 95% confidence limit				
	n=5	n=10	n=20	n=30	n=39	n=5	n=10	n=20	n=30	n=39
5. DNN	0.45	0.22	0.17	0.23	0.30	0.50	0.23	0.18	0.25	0.32
6. DNN \mathcal{L}_{phy}	0.52	0.18	0.15	0.17	0.22	0.59	0.19	0.17	0.18	0.24
7. DNN $^{\text{upd}}$	0.78	0.30	0.12	0.22	0.25	0.81	0.32	0.15	0.24	0.27
8. DNN $^{\text{upd},\mathcal{L}_{\text{phy}}}$	0.78	0.29	0.10	0.18	0.22	0.82	0.33	0.12	0.21	0.25

The 95% prediction intervals (upper limit-lower limit) decrease as increasing amounts of experimental data are used to train the DNN models. However, the prediction intervals obtained using the DNN models are much smaller than the ones obtained using the GP models since the DNN models has more degrees of freedom that can be optimized. For example, for $n = 39$, the prediction interval width is 0.02-0.03 for the DNN models, whereas it is 0.12-0.15 for the GP models. In addition to the larger number of parameters, the number of training epochs, which is the number of complete passes through a batch of training dataset, is optimized for the DNN models, thus maximizing their prediction accuracy. The prediction accuracy is further improved by choosing the appropriate dropout rate as discussed earlier. The number of parameters to be learned reduces with the use of dropout, which helps with regularization and prevents ill-conditioning. Further, the number of training epochs is also observed to affect both the accuracy and uncertainty of the sensitivity estimates. It was found that at a smaller number of epochs, the model was not fully trained resulting in underfitting, leading to larger bias and variance in the prediction. As the number of epochs was increased, both the bias and the variance were reduced.

The results of the numerical example could be summarized as follows:

- The GP models required more computational effort than the DNN models, both in training and prediction.
- The DNN models were able to achieve higher accuracy and lower uncertainty in prediction, due to the optimization of dropout rate and number of training epochs.
- The physics-informed ML models are able to achieve higher prediction accuracy than the basic ML models, especially when the amount of available experimental data is small.

Overall, the DNN models gave higher accuracy and lower uncertainty in the GSA results than

the GP models, and also required less computational effort both in training and prediction.

4.4 Summary

This chapter extended the methodologies for information fusion and machine learning to sensitivity analysis using both physics knowledge and experimental data, while accounting for model uncertainty. Variance-based sensitivity analysis is used to quantify the relative contribution of each uncertainty source to the variability of the output quantity. Two types of ML models were considered, namely, GP and DNN models. Several PIML models were developed by leveraging two strategies for incorporating physics knowledge into ML models: (1) incorporating physics constraints within the loss functions used in training the ML models, and (2) pre-training an ML model with simulation data and then updating it with experimental data. The first strategy does not use the physics model, whereas the second strategy does.

The calculation of the Sobol' indices with the GP model simply uses the proposed estimator (Eq. 4.7). On the other hand, with respect to the DNN model, we use the Monte Carlo dropout strategy to compute prediction bounds on the Sobol' indices; previous work in this regard has only considered prediction bounds of the model output. Prediction bounds are computed for the sensitivity index estimates to account for the model uncertainty in the trained models, and the accuracy and computational effort of the various PIML models are compared.

The results show that the application of PIML strategies to both GP and DNN enables accurate Sobol' index computations even with smaller amounts of experimental data while producing physically meaningful results. Thus, the proposed approach helps to fill the physics knowledge gap in the ML models while estimating the Sobol' indices, by correcting for the approximation in the physics-based models. The numerical examples show that training the GP models and estimating the Sobol' indices require more computational effort than the DNN models. The uncertainty regarding the sensitivity estimates obtained using the DNN models is smaller than the results obtained using the GP models. In the numerical examples, the DNN models are found to be more accurate compared to the GP models. The higher accuracy, lower uncertainty, and lower computational effort of the DNN models is attributed to their flexibility in terms of number of parameters, training epochs and dropout rate.

Adaptive Surrogate Modeling for High-Dimensional Spatio-Temporal Output¹

5.1 Introduction

Chapter 3 discussed time-independent response prediction; even though the process model itself was dynamic, the quantity of interest (QoI) was time-independent. The current chapter focuses on time dependent outputs, where the prediction is not a single value, but a series of values varying over time. The developed methods are illustrated for thermo-mechanical analysis (with spatio-temporal, multivariate output) of a gas turbine blade, but the investigated concepts, such as dimension reduction, surrogate modeling, adaptive sequential sampling etc., are also applicable to other time-dependent systems.

Engineering analyses such as design optimization, uncertainty quantification, reliability assessment, and system health diagnosis and prognosis often require techniques that require multiple runs of the physics model. The construction of an accurate surrogate model requires adequate amount of training data that can be generated by evaluating the physics-based model at multiple settings in the input space. For nonlinear systems with high-dimensional output, such repeated evaluations pose significant challenge w.r.t. computational resources and time, thus motivating the minimization of the number of training runs of the expensive physics-based model.

Several types of surrogate models are used in the literature, e.g., polynomial chaos expansion (PCE) [70], Gaussian process (GP) regression [36], support vector regression (SVR) [113], deep neural networks (DNNs), etc. The first two approaches PCE and GP are computationally demanding in high-dimensional applications. More specifically, estimation of coefficients of a PCE and computing the inverse of the covariance matrix becomes challenging in large dimensions despite recent development of sparse grids [114, 115] and basis adaptive methods [116, 117]. For GP surrogate modeling, in addition to the issue of computing the inverse of the covariance matrix, the number of tuning parameters associated with the correlation function also increases with the dimension, thus limiting the applicability of GPs in high-dimensional applications. As for SVR and DNN and other machine learning models, the accuracy is dependent on the quality and quantity of the training data, which is often limited if the physics model runs are expensive.

Several studies on surrogate modeling techniques have addressed high-dimensional output by

¹Adapted with permission from: Kapusuzoglu, B., Mahadevan, S., Matsumoto, S., Miyagi, Y., & Watanabe, D., “Adaptive Surrogate Modeling for High-Dimensional Spatio-Temporal Output,” (under review), 2022.

mapping the output to the space of principal directions, where the top few components capture most of the variance in the output, using principal components analysis (PCA) [1, 118], or singular value decomposition (SVD) [2, 5, 6]. Co-Kriging [119] has also been used for multivariate outputs, but it can be computationally demanding for high-dimensional field quantities [120]. PCA and SVD are also challenging for very high-dimensional problems since they require large storage and memory for the large covariance matrix. On the other hand, randomized SVD (rSVD) [35, 121–123] offers an efficient way to approximate the dominant singular components in high-dimensional applications, thus allowing for a scalable architecture for modern “big data” applications.

If the surrogate model is constructed in a low-dimensional space, in addition to the surrogate model error in the low-dimensional space, there is also reconstruction error in translating the surrogate model prediction to the high-dimensional original space. Thus, errors need to be quantified in a systematic and useful manner to assess the quality of the surrogate model. However, often the available resources prevent obtaining a large amount of training data when the original model is computationally expensive. Therefore, it is important to construct an accurate surrogate model with the fewest possible experiments or computer simulations. In general, design of experiments (DoE) methods can be classified as being non-adaptive or adaptive. The adaptive DoE techniques consider the physics of the system (as indicated by the previously sampled training points) while sampling new training points. Examples of non-adaptive DoE methods are the full factorial [124], Latin hypercube [17], orthogonal array [125], minimax and maximin-distance designs [126], where the focus is on coverage of the input space and not the physics of the input-output relationship. As a result, the non-adaptive DoE methods may under/oversample and waste computational resources. Whereas, in sequential adaptive sampling, the information obtained from the previous samples is used to populate new samples in regions of high interest, e.g., where the underlying function is highly nonlinear or exhibits abrupt changes and the surrogate model accuracy is poor. Moreover, with adaptive sampling it is possible to stop the computationally expensive sampling process as soon as the surrogate model accuracy reaches an acceptable level.

Two components are essential in identifying additional sampling points to improve the surrogate model: (1) quantification of surrogate model error, and (2) decision criterion or learning function to identify the additional sampling points. Several approaches have been proposed in the literature to address both components. For example, cross-validation (CV) methods use estimates such as Leave-One-Out (LOO) CV errors [127, 128] and new training points are located in the region of the maximum value of the CV error. Adaptive approaches based on cross-validation variance (CVV) select new samples based on the predicted maximum value of the CVV among candidate future sam-

ples [129, 130]. Hombal and Mahadevan [39] proposed selecting training points that focus on minimizing the bias in the prediction. The LOO error-based Accumulative Error (ACE) approach [128] uses a weighted combination of LOO errors. Another method is the MSE method [127, 131] which chooses the next training point based on the maximum value of the estimated mean squared error in the response predicted by a GP surrogate model. The Cross-Validation Voronoi (CVVor) is based on the combination of a cross-validation exploitation with a distance-based exploration, in which a Voronoi tessellation is employed to divide the whole input space into a set of Voronoi cells [132]. The maximin scaled distance (MSD) method [127] is another common method, which is a modification of maximin distance-based sampling that assigns weights to the important variables by using the available information. The Expected Improvement (EI) method uses geometry-based exploration and exploitation based on the variance of the prediction. The main goal of EI is to predict the global minimum value of the response accurately [133]. Another geometry-based exploitation method is Local Linear Approximation (LOLA)-Voronoi, which is a discontinuous adaptive sampling approach based on an exploitation feature with gradient estimation and exploration given by the volume of Voronoi tessellation cells [134]. Adaptive methods using query-by-committee-based exploration such as Mixed Adaptive Sampling Algorithm (MASA) [135] are studied in the literature, where new sample point is found by combining a local exploitation contribution based on Query by Committee (QBC) fluctuation and a global exploration based on distance.

Adaptive sequential sampling design (also referred to as active learning [136, 137]) has also been investigated for reliability analysis, i.e., for the estimation of probability of failure; the focus of such studies has been accurate surrogate modeling of the limit state, often formulated as $g() = 0$, which is the boundary between regions of success and failure. The efficient global reliability analysis (EGRA) [138] method combines efficient global optimization (EGO) [133] with a learning function to adaptively select training points, in the context of Gaussian process (GP) surrogate modeling of the limit state. Another active learning method for reliability analysis is AK-MCS [139] that uses another learning function to adaptively select new training points for the GP model improvement. In order to improve the speed of convergence of the AK-MCS method, a widely used learning function U was proposed, resulting in the AK-MCS + U method [140]. Based on the AK-MCS + U method, Hu and Mahadevan [141] proposed an enhanced surrogate model-based reliability analysis method based on global sensitivity analysis to further improve the computational efficiency. All of the above-mentioned methods focus on approximating only the limit state $g() = 0$, whereas our concern in this chapter is with a general prediction model, i.e., predicting the output accurately over the entire input space.

Overall, the limitations of existing surrogate modeling methods can be summarized as follows: (a) looking at a single limit state instead of the entire input space, (b) considering only scalar output, and (c) inability to handle high-dimensional systems. In order to address these limitations an adaptive surrogate modeling strategy is developed in this chapter to predict the high-dimensional spatial and temporal output quantities of interest (QoIs).

The proposed approach addresses two challenges: high-dimensionality of the output and adaptive selection of training runs for the surrogate model. For dimension reduction, the commonly used SVD is not feasible for very high-dimensional problems due to the need to store and invert very large correlation matrices. Instead, the method of randomized singular value decomposition (rSVD) [35] is found to be effective for problems with very high-dimensional output, and is used to identify the important features in the output space to give a lower dimensional representation of the original QoIs. These important features are then used to construct the surrogate model in the low dimensional space. Subsequently, the trained model is used to predict the QoIs in the original space. Error analysis is performed both in the lower dimensional latent space and the high dimensional original space. This helps to quantify the contributions of surrogate model error and reconstruction error separately. This lays the foundation for adaptive improvement of the surrogate model. A novel approach that combines the ideas of exploration and exploitation for adaptive training point selection was developed in a manner that is applicable to time-dependent multi-physics dynamic problems with high-dimensional spatio-temporal outputs.

The important contributions of this chapter can be summarized as (1) Dimension reduction for high-dimensional spatio-temporal outputs; (2) Surrogate model error quantification in two spaces; and (3) New sequential sampling technique for adaptive surrogate model improvement for multivariate spatio-temporal data.

5.2 Proposed Methodology

The proposed methodology aims to efficiently construct a surrogate model for high-dimensional spatio-temporal response prediction. In this section, we first discuss dimension reduction to obtain a lower dimensional representation, features, where we adapt the rSVD method for high-dimensional spatio-temporal output using a two-step mapping strategy. Next, we investigate multiple options for surrogate modeling and select the best surrogate model for a multivariate output. This is achieved by performing a cross-validation analysis for each multivariate surrogate model constructed in a low-dimensional space for the important features. Following this, we develop the adaptive sampling strategy for identifying additional training points to improve the surrogate model, by combining

exploitation and exploration. Thus the proposed overall methodology consists of four steps:

1. Dimension reduction (two-step mapping)
2. Surrogate model construction
3. Surrogate model error quantification (through cross-validation)
4. Adaptive training point selection

The following subsections describe these steps in detail.

5.2.1 Dimension Reduction

The rSVD method described in Section 2.1.2 is extended in this work for dimension reduction. The following additional steps are proposed to adapt this method for a high-dimensional spatio-temporal output, in a manner that facilitates surrogate model construction.

A two-step dimension reduction approach is employed to identify the important features in the output space to give a lower dimensional representation of \mathbf{S} . The dimension of the output data matrix $\mathbf{S} \in \mathbb{R}^{(n_t \times n_d) \times n_s}$ has a large number of features and it has more features (columns) than observations (rows). Thus, in the first step of the two-step approach the original features of the output data matrix can be reduced using rSVD to a smaller subset of features that are most relevant to the prediction problem. The result is matrix $\mathbf{I}^{(1)} \in \mathbb{R}^{(n_t \times n_d) \times n_1}$ with a lower rank n_1 that is said to approximate the original spatio-temporal output \mathbf{S} . Let us denote the truncated singular value matrix, eigenvector matrix, and orthogonal matrix obtained using n_1 important features as $\mathbf{U}^{(1)} \in \mathbb{R}^{(n_t \times n_d) \times n_1}$, $\Sigma^{(1)} \in \mathbb{R}^{n_1 \times n_1}$, $\mathbf{V}^{(1)} \in \mathbb{R}^{n_1 \times n_1}$. The compressed form of $\mathbf{I}^{(1)}$ is $\mathbf{I}^{(1)} \approx \mathbf{U}^{(1)} \Sigma^{(1)} (\mathbf{V}^{(1)})^T$. The resulting matrix \mathbf{I}^1 still has time-dependency. Therefore, in the second step of the two-step approach the time-dependency needs to be removed. When rSVD is performed twice on the transformed version of matrix $\mathbf{I}^{(1)}$, i.e., $\mathbf{I}^{(1)'} \in \mathbb{R}^{n_d \times (n_1 \times n_t)}$, the columns of this low-rank matrix $\mathbf{I}^{(2)}$ would represent the important features (i.e., the number of features n_2 to build the surrogate model) and the rows represent the model parameters of FEM simulations. Thus, we obtain $\mathbf{I}^{(2)} \approx \mathbf{U}^{(2)} \Sigma^{(2)} (\mathbf{V}^{(2)})^T$, where $\mathbf{U}^{(2)} \in \mathbb{R}^{n_d \times n_2}$, $\Sigma^{(2)} \in \mathbb{R}^{n_2 \times n_2}$, $\mathbf{V}^{(2)} \in \mathbb{R}^{n_2 \times (n_1 \times n_t)}$.

The accuracy of the above two-step dimension reduction needs to be evaluated. To do this, the low-rank approximated matrix $\mathbf{I}^{(2)}$ in the second feature space is mapped to the original space (i.e., first it is mapped to the first feature space and then mapped to the original space). The reconstruction accuracy is quantified using root mean square error (RMSE). The RMSE value decreases with the number of important features used to approximate the original matrix \mathbf{S} . The number of

important features is chosen based on the reconstruction accuracy and the percentage of variance explained by the top few singular vectors and singular value pairs. The proposed two-step dimension reduction approach is used in the numerical example in Section 5.5. Next, the surrogate model construction described in Section 5.2.2 is performed in the second low-dimensional space using the important features $\xi^{(2)} = \mathbf{U}^{(2)}\Sigma^{(2)}$.

5.2.2 Surrogate Model Construction

Any of the available surrogate modeling techniques mentioned earlier (e.g., PCE, GP, SVR, DNN etc.) can be used to construct the surrogate model. In addition to these techniques, other prominent machine learning algorithms based on boosting (which is one of ensemble learning algorithms) are also available for improving the performance of a simple machine learning model. This paper explores several of them such as Light Gradient Boosting Machine (lightGBM) [142], Extreme Gradient Boosting (XGBoost) [143], Categorical Boosting (CatBoost) [144], and random forest (RF) [145]. Gradient Boosting Decision Tree (GBDT) is a relatively new decision tree-based ensemble learner. LightGBM, XGBoost, and CatBoost are different variations of gradient boosting methods. The main difference between these techniques is how they build the decision tree.

A tree-based regression technique known as extremely randomized tree regressor or simply Extra-Trees regressor [146] is found to give the best performance in the numerical example (Section 5.5), based on comparing the k-fold cross validation error results for various techniques. Tree-based ensemble methods randomly construct more than one decision tree to achieve an increase in the generalization performance. For example, random forest (RF) regression [145] trains the trees with bootstrap samples for each candidate split based on randomly selected subset of the features. The main idea behind Extra-Trees is to randomly create a number of different trees with randomly chosen features. The randomization helps to achieve a greater reduction in the variance of the model prediction, compared with other ensemble methods like RF [145]. The main differences between the Extra-Trees algorithm and RF regression are: (a) Extra-Trees randomly splits nodes using a random subset of the features selected at every node, rather than the best split used in RF; and (b) RF applies the bagging procedure to iteratively generate sub-training sets (bootstrap samples) with replacements, while Extra-Trees uses all the training samples to construct each tree with varying numbers of parameters.

5.2.3 Surrogate Model Error Quantification

The surrogate model prediction error is used to inform about areas with the highest error. The Leave-One-Out Cross-Validation (LOOCV) method is a special case of the k -fold cross-validation (CV), with $k = N$. For each observation $i \in [1, N]$, a separate surrogate model \mathcal{M}_{-i} is trained on $N - 1$ observations consisting of the reduced set $\mathcal{D}_{-i} = \mathcal{D} \setminus (\mathbf{x}^i, \mathbf{y}^i)$. The surrogate model accuracy is then evaluated on the test point \mathbf{x}^i . The spatio-temporal physical quantities are predicted for the test data using the surrogate model trained in the latent space, where high-dimensional spatio-temporal outputs are projected to. The surrogate model outputs in the feature space are then mapped to the original space (reconstruction) to evaluate the surrogate model accuracy in the original space.

Different error metrics are suitable in different situations. For example, an engineer may be more interested in the relative error of creep damage response in the regions where strong creep behavior is expected since a small amount of change can lead to detrimental effects. Whereas the engineer is less interested in the relative error of the predicted response in regions where weak creep behavior (i.e., physical quantities are close to zero) is dominant. Thus, for this purpose, two error metrics are calculated using LOOCV, namely mean absolute error (MAE) and relative mean absolute error (rMAE). The MAE metric is defined as follows:

$$\text{MAE}_j = \text{mean}(\text{AE}_j) = \frac{1}{n_{\text{time}}} \sum_i^{n_{\text{time}}} |S_{i,j} - \hat{S}_{i,j}| \quad (5.1)$$

where the subscript j refers to the j th FEM node (e.g., MAE_j is the mean absolute error of j th node, where the mean is taken across time for each spatial location). The main challenge of the MAE metric is the identification of thresholds. Moreover, the MAE metric is scale-dependent. In order to address such limitations, the relative mean absolute error (rMAE) metric that is scale independent and less sensitive to outliers is investigated. The rMAE metric is defined as follows:

$$\text{rMAE} = \frac{\text{MAE}}{\text{mean}|\mathbf{S} - \bar{\mathbf{S}}|} \quad (5.2)$$

where $\bar{\mathbf{S}}$ is the mean of the time series output at each spatial location. An advantage of this method is its interpretability. For example, rMAE measures the possible improvement of the proposed model relative to the benchmark model (i.e., the denominator of Eq. (5.2)). When rMAE is less than 1, the proposed model is better than the benchmark model, and when the value of this error metric is greater than 1, then the proposed model is worse than the benchmark model. The only circumstance under which rMAE would be infinite or undefined is when all historical observations are equal (i.e.,

static problem).

For each node in the FEM model, the error metrics are evaluated based on the LOOCV predictions. Note that the surrogate model predicts features in the low-dimensional latent space, which are then mapped to the original space to evaluate the error metrics for the spatio-temporal physical outputs. Then, PNMAE, the percentage of nodes having MAE values greater than some threshold value, and PNrMAE, the percentage of nodes having rMAE values greater than one, are calculated for the current training points using LOOCV. However, there is no way to assess these metrics for points not in the current design, i.e., unobserved points (or candidate new points, \mathbf{x}^c). Therefore, additional surrogate models are built to learn the relationship between the training points and the corresponding LOOCV prediction error metrics in the original space. In this study, Gaussian process (GP) surrogate models are used (allowing the use of Expected improvement (EI) [133]) to predict e_{PNMAE} and e_{PNrMAE} for unobserved points. These GP models are developed using the initial training points. As new points are added to the design \mathcal{D} , the training data used to train these GP models are updated.

5.2.4 Proposed Adaptive Sample Selection Method

First, the approach generates an initial design. The initial design is chosen using a space-filling criterion with a given initial number of points. In this work, we use maximin Latin hypercube sampling (LHS) [147–149], though any other space-filling metric can be used, to generate initial training points that cover the input space uniformly. Based on this initial design, the LOOCV error e_{LOOCV} is calculated for each point in the design space by building a surrogate model each time leaving one point out. A secondary GP surrogate model is constructed for e_{LOOCV} to predict e_{LOOCV} for candidate points for additional samples.

A straightforward approach is to simply use the LOOCV error for adaptive sampling, i.e., exploit regions with high prediction error to select new training points. However, a purely exploitation-based approach can result in clustered samples in the input space. In order to avoid this problem, an exploration strategy based on a space-filling criterion can be added to spread the new samples over the entire input domain while exploiting the regions of interest.

Thus two strategies, namely exploitation and exploration, are combined here for surrogate model improvement through adaptive selection of additional training points. Exploration aims to discover regions of the input space not covered in previous training points in order to obtain knowledge of the response over the entire design space; thus exploration does not use the simulation outputs for the previous training points. Whereas exploitation uses the output information gained from

previous training points to identify high-interest subregions to generate new samples in the vicinity of these regions. These subregions can be associated with large prediction error, significant non-linear behavior, discontinuity, etc. New samples can be added in the region of interest depending on the aim of the analysis to build a surrogate model that predicts the response accurately in the exploited high-interest regions. Instead of considering these strategies individually, it can be beneficial to consider them together in hybrid adaptive learning to leverage both of their strengths, such that exploration adds points from previously unexplored regions (geometry-based) and exploitation adds points in regions of interest pertaining to surrogate model accuracy (physics-based).

5.2.4.1 Space-Filling Criterion

A space-filling criterion is used to avoid the clustering of samples that could occur in a purely exploitation-based approach and ensure uniform coverage of the design space. The space-filling criterion used in this work is based on Euclidean distance, specifically the maximin distance, wherein the minimum non-zero distance d_{min} of a candidate point from all other points in the current set of training points is computed. A candidate point (\mathbf{x}^c) with the maximum of these minimum distances is selected as the new training point from \mathbf{x}^* . Thus, the exploration strategy is given by:

$$\begin{aligned} \mathbf{x}^c &= \arg \max_{\mathbf{x}^* \in \mathbb{X}} (d_{min}) \\ d_{min} &= \min_{\substack{\mathbf{x}^i \in \mathcal{D} \\ \mathbf{x}^* \in \mathbb{X}}} \|\mathbf{x}^i - \mathbf{x}^*\| \end{aligned} \quad (5.3)$$

5.2.4.2 Expected Improvement

In order to overcome the overexploitation problem, which could cause new samples to be clustered in regions with large mean LOOCV prediction error, a well-known approach is to use the Expected improvement (EI) [133] function, which helps to trade-off between exploitation and exploration. It can be expressed as

$$EI(\mathbf{x}) = \begin{cases} (\mathbf{y}^* - \mu(\mathbf{x})) \Phi(u) + \sigma(\mathbf{x}) \phi(u), & \text{if } \sigma(\mathbf{x}) > 0. \\ 0, & \text{if } \sigma(\mathbf{x}) = 0. \end{cases} \quad (5.4)$$

where $u = (\mathbf{y}^* - \mu(\mathbf{x})) / \sigma(\mathbf{x})$, \mathbf{y}^* being the current best solution chosen from among the true function values at the training points and $\phi(\cdot)$ and $\Phi(\cdot)$ represent the PDF and CDF of the standard normal distribution, respectively. EI is a non-negative, parameter-free function and is zero at the training points.

5.2.4.3 Proposed Learning Function

Combining the EI function defined in Section 5.2.4.2 and the space-filling criterion described in Section 5.2.4.1, a point with the largest expected improvement EI_{PNMAE} (i.e., expected improvement based on PNMAE) and EI_{PNrMAE} (i.e., expected improvement based on PNrMAE) and the maximum d_{\min} is selected as the new sample point. Thus, a learning function that facilitates the active learning process is proposed here as:

$$\mathcal{L}(\mathbf{x}) = (\beta_1 EI_{\text{PNMAE}} + \beta_2 EI_{\text{PNrMAE}})^\theta \times (d_{\min}(\mathbf{x}))^\gamma \quad (5.5)$$

where $\beta_1, \beta_2 \in [0, 1]$ control the relative contributions of EI_{PNMAE} and EI_{PNrMAE} respectively, and $\alpha, \gamma \in [0, 1]$ are used to control the trade-off between exploitation and exploration respectively. (Other mathematical formulations of the learning function using these metrics are also possible). In this formulation of the learning function, EI provides both exploitation and exploration based on the mean and variance of the prediction error, whereas the space-filling criterion only provides geometry-based exploration. The two extreme cases, $\alpha = 0$ and $\alpha = 1$, respectively denote that exploitation is not considered and exploitation is considered. Exploration of the input space is controlled by γ . When $\gamma = 0$, the minimum Euclidean-distance is not considered (i.e., no exploration), whereas the larger the γ value is, the higher the weight of the exploration term. Note that \mathcal{L} provides a trade-off between the space-filling criterion, which avoids clustering, and the largest estimation of prediction error.

The parameters of the learning function can be estimated by using an algorithm similar to the Expectation-Maximization (EM) algorithm [150]. We have 2 sets of parameters. First, we fix the first set $\{\beta_1, \beta_2\}$ and compute the second set $\{\alpha, \gamma\}$ in one step. Then, we compute the first set based on the computed values of the second set in the previous step. These two steps are repeated until the maximum change in the parameter estimates between consecutive iterations does not exceed a convergence criterion.

The new sample point \mathbf{x}^c that results in the maximum learning function value \mathcal{L}_{\max} is identified as follows:

$$\begin{aligned} \mathcal{L}_{\max} &= \max_{\mathbf{x}^* \in \mathbb{X}} [\mathcal{L}(\mathbf{x}^*)] \\ \mathbf{x}^c &= \arg \mathcal{L}_{\max} \end{aligned} \quad (5.6)$$

Note that the evaluations of candidate points using the proposed method are computationally

inexpensive since these evaluations are based on the current surrogate model, not the original expensive physics model.

5.2.4.4 Stopping Criteria

The computational resource limit (i.e., number of training points, N), and the LOOCV error can be used as the stopping criteria. The latter criterion is used to evaluate the surrogate model performance on the test set. The surrogate model accuracy is evaluated as new samples are added to the design based on the proposed adaptive sampling approach. If the LOOCV error metrics e_{PNMAE} and e_{PNrMAE} achieve an acceptable value, then stop adding new points. In this work, both the computational resource limit and surrogate model accuracy are used as the stopping criteria. The detailed sequence of steps of the proposed adaptive learning approach is outlined in Algorithm 3.

Algorithm 3 Identifying new training points for adaptive improvement of the surrogate model

Input: \mathcal{D}

Output: \mathbf{x}^c

- 1: Generate an initial design using maximin LHS
 - 2: procedure Output Dimension Reduction
 - 3: Construct the data matrix, \mathbf{S}
 - 4: Perform a two-level rSVD on the matrix \mathbf{S} to obtain important features
 - 5: end procedure
 - 6:
 - 7: procedure Leave-One-Out Cross-Validation (LOOCV)
 - 8: Split the design set \mathcal{D} into k disjunct sets \mathcal{D}_i , $i = 1, \dots, k$, with $k = N$ (LOO)
 - 9: for $i = 1$ to N do
 - 10: Train an Extremely Randomized Trees machine learning model \mathcal{M}_{-i} with $N - 1$ features consisting of the reduced set $\mathcal{D}_{-i} = \mathcal{D} \setminus (\mathbf{x}^i, \mathbf{y}^i)$.
 - 11: Calculate e_{PNMAE} and e_{PNrMAE} for the current training points in the reduced set \mathcal{D}_{-i} :
 - 12: $e_{\text{PNMAE}} = \%$ of nodes having MAE $\geq 2.5\text{e-}4$
 - 13: $\text{MAE}_j = \text{mean}(\text{AE}_j) = \frac{1}{n_{\text{time}}} \sum_i^{n_{\text{time}}} |\mathcal{S}_{i,j} - \hat{\mathcal{S}}_{i,j}|$
 - 14: $e_{\text{PNrMAE}} = \%$ of nodes having rMAE ≥ 1
 - 15: $\text{rMAE} = \frac{\text{MAE}}{\text{mean}|\mathbf{S} - \hat{\mathbf{S}}|}$
 - 16: end for
 - 17: end procedure
 - 18:
 - 19: procedure Adaptive Learning
 - 20: while Average PNrMAE $\leq 15\%$ or $N \geq 66$ do
 - 21: Construct GP models for e_{PNMAE} and e_{PNrMAE} (LOOCV errors) to make predictions \hat{e}_{PNMAE} and \hat{e}_{PNrMAE} for candidate points
 - 22: Define the learning function:
 - 23: $\mathcal{L} = [(\beta_1 E I_{\text{PNMAE}} + \beta_2 E I_{\text{PNrMAE}})^\alpha \times (d_{\min}(\mathbf{x}^*))^\gamma]$
 - 24: Sample a large set of random points from the design space and evaluate \mathcal{L} for these points
 - 25: Identify the largest learning function value; $\mathcal{L}_{\max} = \max \mathcal{L}(\mathbf{x}^*)$
 - 26: Choose the new training point \mathbf{x}^c that results in \mathcal{L}_{\max} , i.e., $\mathbf{x}^c = \arg \mathcal{L}_{\max}$.
 - 27: end while
 - 28: end procedure
-

5.3 Summary of Methodology

The proposed methodology for adaptive surrogate modeling with high-dimensional spatio-temporal output as shown in Fig. 5.1 consists of the following steps:

1. Construct the output data matrix \mathbf{S} based on the available simulations where each row represents a time instant corresponding to one FEM run, and the columns represent multiple spatial locations of multiple output quantities for that FEM run;
2. Perform the two-step dimension reduction approach explained in Section 5.2.1 to find the lower dimensional representation $\mathbf{I}^{(2)}$;
3. Build a surrogate model for each feature vs. process inputs in the feature space;
4. For the test set points, project the surrogate model prediction in the feature space back to the original space to obtain the predictions in the original space, $\hat{\mathbf{S}}$;
5. Use cross-validation to evaluate the surrogate model accuracy in the original space;
6. If the accuracy of the surrogate model is not acceptable, then use the proposed adaptive sequential sampling based on the learning function given in Section 5.2.4.3 to propose additional training points for the surrogate model;
7. Repeat steps 1 to 6 until the model accuracy reaches an acceptable value or until the computational resource limit reached.

5.4 Evaluation of Proposed Approach using Benchmark Problems

In this section, the proposed adaptive sampling technique is compared to several existing techniques described in Section 5.1 (i.e., EI, CVV, LOLA, MSD, MASA, CVVor). To provide a fair comparison, the performance of each sampling technique in accurately capturing seven benchmark test functions of different complexity is analyzed (see Appendix A.1). Normalized root mean square error (NRMSE = $\text{RMSE}/(y_{\max} - y_{\min})$), where $\text{RMSE} = \sqrt{\sum_{i=1}^n (y_i - \hat{y}_i)^2/n}$ is first computed for each fold of LOOCV, followed by the computation of the average LOOCV error, by taking the average across all the LOOCV folds. This average LOOCV error is used to assess the performance of all the adaptive sampling techniques.

Adaptive sampling techniques with higher exploitation component are strongly dependent on the size of the initial sample. A few empirical formulas have been proposed in the literature for choosing the initial sample size [133, 151]. In this work, the number of samples included in the initial design

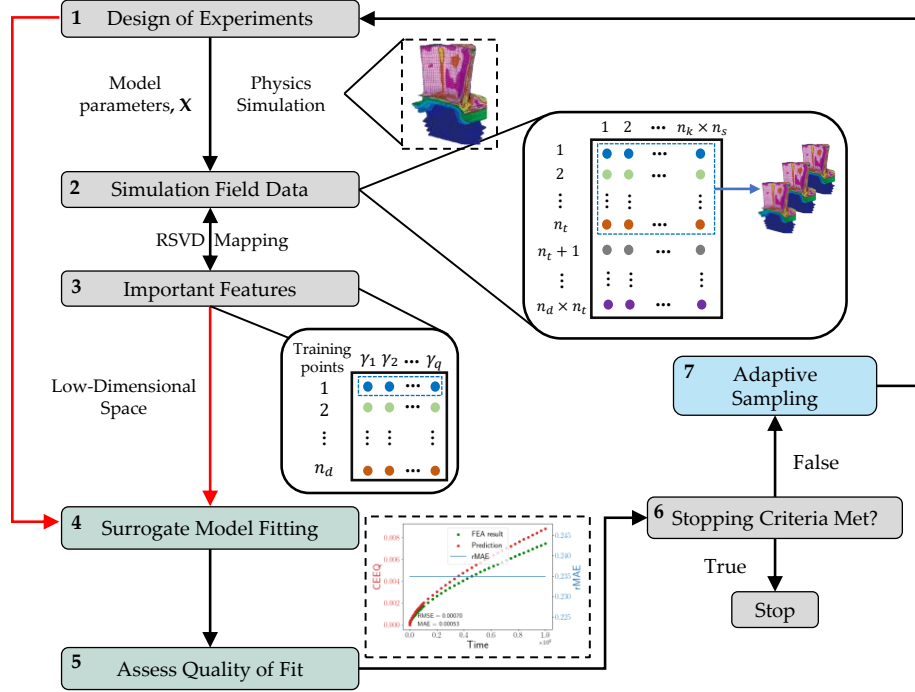


Figure 5.1: Schematic describing the proposed procedure for surrogate model construction with high-dimensional field data using rSVD

(m) is chosen based on the rule $m = 10n$ [133], where n is the dimension of the input space. For each two-dimensional test function in Appendix A.1, an initial training set consisting of 20 points is generated using maximin LHS and used to start all the adaptive sampling techniques. The positions of the 20 initial sample points are shown by black dots in Figs. 5.3 and 5.4. In order to have a fair comparison between the proposed method and the existing adaptive sampling techniques, a Gaussian process (GP) regression model is built with the exact same properties in each fold of LOOCV-based techniques. Then the number of training points is increased up to 45 by adding a new point in each step for each adaptive sampling technique, following each technique's procedure. The NRMSE results for different sampling techniques for 45 total samples are shown in Fig. 5.2 for different benchmark functions. For illustration purposes, the adaptively selected samples are highlighted by red dots (larger points indicate a sample very close to a neighboring point) and the corresponding surrogate model accuracy in terms of NRMSE are shown as contours in Figs. 5.3 and 5.4 for the Branin and Schwefel functions in Appendix A.1 respectively, for each sampling technique.

The Expected Improvement Cross-Validation (EICV) and Space-Filling Expected Improvement Cross-Validation (SFEICV) techniques are special cases of the proposed method; these two methods are compared against other adaptive improvement methods in this section. In EICV, $\gamma = 0$, therefore

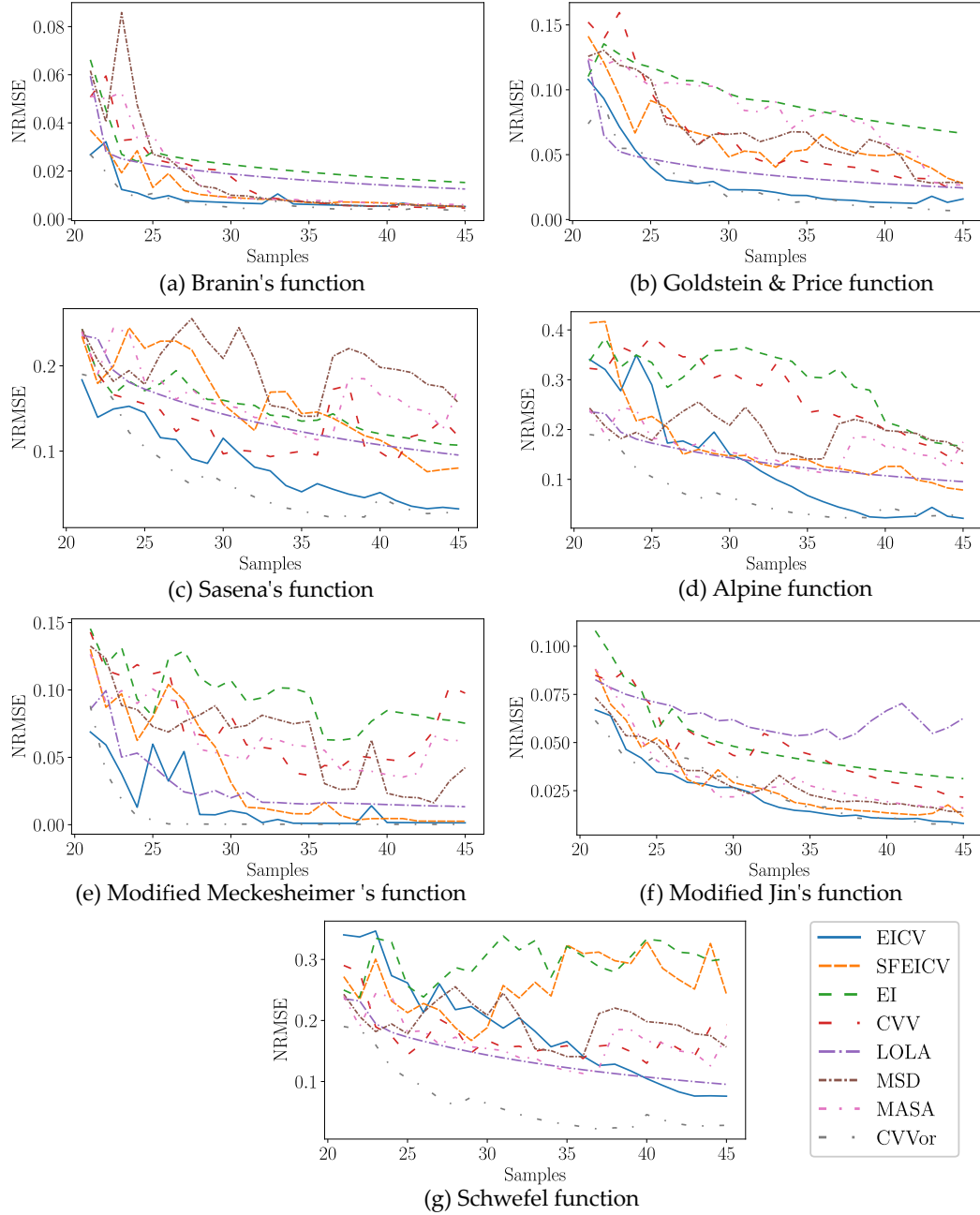


Figure 5.2: Evolution of the NRMSE from the initial surrogate model (20 samples) to the final surrogate model (45 samples) using different adaptive sampling techniques, for the seven benchmark functions in Appendix A.1

the weighted minimum Euclidean distance is not considered (i.e., no additional exploration other than the exploration already inherent in EI). Whereas in SFEICV both θ and γ equal 1, thus the weight of exploration is higher. (Note that two separate GP models are constructed for EICV and SFEICV, where the first GP model is to predict the original output QoI, and the second GP model

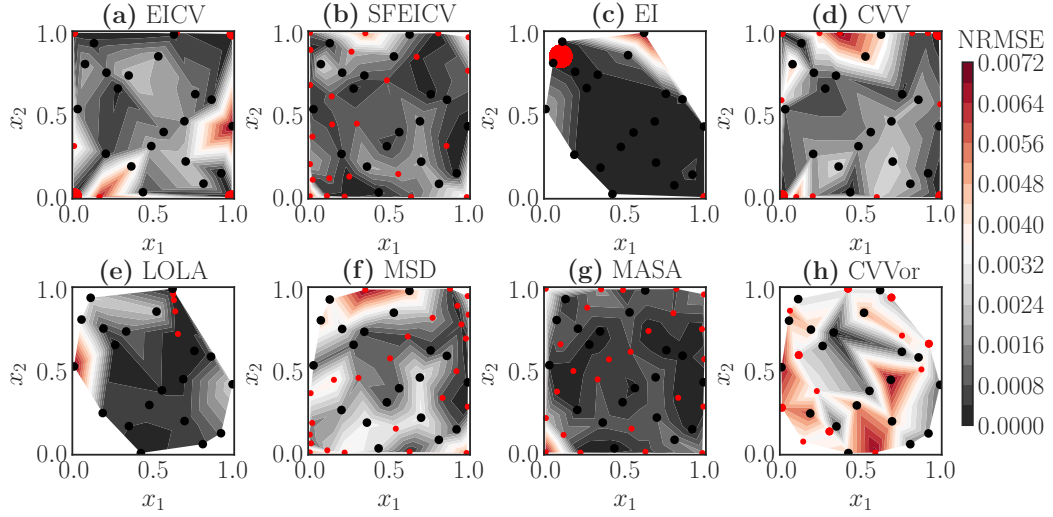


Figure 5.3: Adaptively selected training point locations for the Branin’s function and the final surrogate model accuracy in terms of NRMSE are plotted in contours (Black dots: initial 20 samples, Red dots: additional 25 samples)

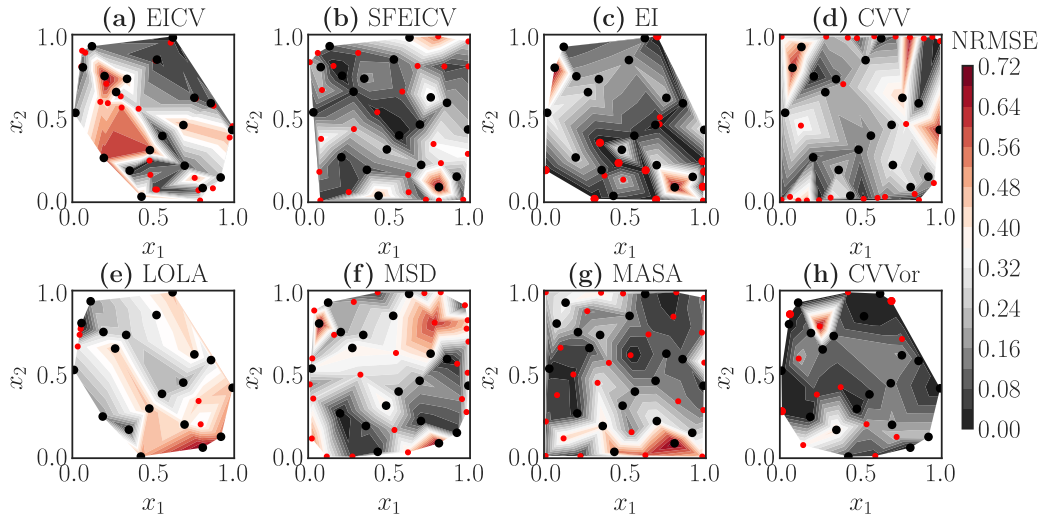


Figure 5.4: Adaptively selected training point locations for the Schwefel function and the final surrogate model accuracy in terms of NRMSE are plotted in contours (Black dots: initial 20 samples, Red dots: additional 25 samples)

is to predict the LOOCV error as described in Section 5.2.4).

An issue with some techniques with the exploitation component is that they may focus on insignificant features with respect to the problem of interest. This problem appears for EI, where it is originally designed [133] to estimate the minimum of the response surface as shown in Figs. 5.3(c) and 5.4(c). EI can over-exploit in many situations, such as when it hits a local optimum (see Figs. 5.3(c) and 5.4(c)). When this is the case, EI cannot capture the proper behavior of the entire function

and its performance becomes highly dependent on the initial dataset. In most complex engineering problems, the initial dataset is limited, thus it is important to select an adaptive technique such as the proposed EICV that does not focus on insignificant features of the problem and exploits the regions of interest. The gradient estimation approach used in Local Linear Approximation (LOLA)-Voronoi gets increasingly complex as the problems become high-dimensional and complex. The performance of LOLA is also not as good as EICV (see Figs. 5.2(a)-(g)), which is a more robust and computationally efficient technique for high-dimensional problems. It should be noted that all adaptively selected samples (i.e., red dots in Figs. 5.3 and 5.4) using the LOLA technique are in regions where the initial surrogate model error was already low. This behavior of the LOLA technique can be observed in Fig. 5.3(e). The computational complexity of Mixed Adaptive Sampling Algorithm (MASA) depends on the number of committee members (e.g., in the context of GP models, committee members could be autocorrelation functions). In these test problems, three GP models with different autocorrelation functions (Matérn, squared exponential kernel with a different correlation length for each coordinate, and squared exponential kernel with same correlation length in all coordinates) are considered as committee surrogates (e.g., GP models based on different autocorrelation functions). MASA shows a higher emphasis on exploration (see Figs. 5.3(g) and 5.4(g)) and is not as dependent on the initial sample size as some other methods such as EI and MSD. The Maximin scaled distance (MSD) method performs poorly because the method is not capable of exploiting as much as the other methods (as seen in Figs. 5.2, 5.3 and 5.4). SFEICV is almost as good as MASA and MSD at exploring the design space (see Figs. 5.3(b) and 5.4(b)), and it can also exploit the regions where there is high prediction error due to local non-linearity (note the additional samples selected on the bottom left corner in Fig. 5.3(b)).

It should be noted that Cross-Validation Voronoi (CVVor) and EICV yield the most complete performance and best accuracy across all the investigated test cases (see Fig. 5.2). They achieve the smallest NRMSE values, close to zero for all test functions, with the updated surrogate model (i.e., updated with 25 samples). An important difference between these adaptive techniques is the computational effort. The computational effort required by methods based on the LOOCV error such as CVVor or EICV is relatively higher, especially as the dimensions increase. However, they yield reliable results and require less development and user-knowledge. Depending on the complexity and dimension of the problem, the SFEICV technique may be useful in the first few adaptively selected samples, but as more samples are collected, the EICV method becomes more promising. This can be seen in Figs. 5.2(a)(d)(e)(f) where SFEICV yields similar NRMSE values to EICV after 10 adaptively selected training points, i.e., 30 total training points, whereas, EICV results in better

improvement in the surrogate model accuracy after 30 samples.

It should be noted that Cross-Validation Voronoi (CVVor) and EICV yield the most complete performance and best accuracy across all the investigated test problems (see Fig. 5.2). They achieve the smallest NRMSE values, close to zero for all test functions, with the updated surrogate model (i.e., updated with 25 samples). An important difference between these adaptive techniques is the computational effort. The computational effort required by methods based on the LOOCV error such as CVVor or EICV is relatively higher, especially as the dimensions increase. However, they yield accurate results and require less development and user-knowledge. Depending on the complexity and dimension of the problem, the SFEICV technique may be useful in the first few adaptively selected samples, but as more samples are collected, the EICV method becomes more promising. This can be seen in Figs. 5.2(a)(d)(e)(f) where SFEICV yields similar NRMSE values to EICV after 10 adaptively selected training points, i.e., 30 total training points, whereas, EICV results in better improvement in the surrogate model accuracy after 30 samples.

Note that two separate GP models are constructed in EICV and SFEICV (which are special cases of the general method proposed in Section 5.2.4.3), thus they require more computational effort than the other adaptive sampling techniques that are not based on the LOOCV error. However, they do not require the tessellation of the input space into Voronoi cells, which also requires high effort, as in CVVor and LOLA. In addition, the accuracy of EICV and SFEICV is better or comparable to the other methods for different benchmark functions.

5.5 Engineering Application

A simplified gas turbine engine blade model is studied in this section to demonstrate the proposed surrogate modeling approach for an engineering application with high-dimensional spatio-temporal output. Six output quantities from the physics model (creep equivalent strain (CEEQ), creep damage (Dc), von Mises stress (Mises), and displacements in x, y, and z directions) are of interest, as shown in Table 5.1; these are available at a large number of spatial and temporal points. The input space is four-dimensional and consists of turbine blade coating thickness (TBC thickness), turbine output rate (T1T rate), and transient and constant creep rates. The ranges of the inputs are: TBC thickness $[2/3 \cdot \text{nominal}, 4/3 \cdot \text{nominal}]$, T1T rate $[80 \cdot \text{nominal}, 100 \cdot \text{nominal}]$, transient and constant creep rate $[-3\sigma, 3\sigma]$. These model inputs are considered as uncertain. The boundary conditions are assumed to be known with certainty. The total number of nodes is 29,374 for each output quantity of interest (QoI) and the total number of time steps is 54. Thus, the problem has over 176,244 spatial dimensions (i.e., $29,374 \times 6$) in the output space for a single FEM simulation. Considering

also the temporal dimension, the problem dimension is over 9,000,000 (i.e., $176,244 \times 54$) just for a single FEM simulation.

Table 5.1: Physics model input and output parameters

Input	Values
TBC thickness	$\frac{2}{3} \times \text{nominal}$ to $\frac{4}{3} \times \text{nominal}$
TIT rate	$80 \times \text{nominal}$ to $100 \times \text{nominal}$
Transient creep rate	-3σ to 3σ
Constant creep rate	-3σ to 3σ
Output	Nomenclature
Creep equivalent strain	CEEQ
Creep damage	Dc
Von Mises stress	Mises
X displacement	X
Y displacement	Y
Z displacement	Z

5.5.1 Generation of Initial Training Data

The initial surrogate model is built with 46 training points (from 46 FEM runs). Prior to this study, nine initial input settings had been generated using Taguchi L9 orthogonal design [152] for exploratory analysis. To this set of 9 points, we added 37 points, generated using maximin Latin hypercube sampling (LHS). These 46 sets of inputs and the corresponding FEM outputs are then used to build the initial surrogate model.

In order to obtain the maximin LHS for the 4-dimensional input space, 100 LHS designs are generated. Based on the spread of the training points in the design space, the design with the highest spatial coverage is selected as the optimum. To quantify how spread out the points are in each design set, the sum of pair-wise Minkowski distances when $p = 1$ within the set is selected as a metric. For two row vectors, $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^n$ in the matrix, the Minkowski distance is defined as:

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^n |\mathbf{x}_i - \mathbf{y}_i|^p \right)^{(1/p)}. \quad (5.7)$$

The higher the $d(\mathbf{x}, \mathbf{y})$ is, the higher the occupancy rate (coverage) in the design space. The sum of the pair-wise Minkowski distances was calculated for each LHS design and the optimum design is selected to give the 37 additional training points for the initial surrogate model.

5.5.2 Dimension Reduction of the Output Space

Based on the correlation analysis, CEEQ and Dc are found to be perfectly correlated with each other at all time steps [153]. Similarly, Y and Z displacements are negatively correlated, and the correlation becomes more significant at higher time steps. In order to take this correlation between

the output QoIs into account and improve the computational efficiency, dimension reduction of the original output matrix is performed using rSVD for all 6 QoIs and then a single surrogate model, Extra-Trees regressor, is built to predict multivariate time series for all 6 QoIs. To be able to jointly predict the QoIs at each time step for untested configurations, the output data matrix \mathbf{S} is constructed as follows:

$$\mathbf{S} = [\mathbf{D}_1 \mathbf{D}_2 \mathbf{D}_3 \mathbf{D}_4 \mathbf{D}_5 \mathbf{D}_6]_{(n_{time} \times n_{sim}) \times (n_{nodes} \times n_{QoI})} \quad (5.8)$$

where $\mathbf{D}_i \in \mathbb{R}^{n_{time} \times n_{nodes}}$, $i = (1, 2, \dots, n_{QoI})$ is the i th QoI data matrix.

The matrix $\mathbf{S} \in \mathbb{R}^{(n_{sim} \times n_{time}) \times (n_{nodes} \times n_{QoI})}$ is constructed such that the number of rows is equal to the total number simulations (n_{sim}) times the total number of time steps of each simulation (n_{time}), and the number of columns is equal to number of nodes (n_{nodes}) times the total number of QoIs (n_{QoI}). For example, for the final surrogate model we have output data for 66 simulations and the output data matrix $\mathbf{S}_{(54 \times 66) \times (29374 \times 6)}$ consists of approximately 629 million data points.

The reconstruction accuracy is evaluated using RMSE. The percentage of variance explained by the top 50 and 20 features in the first and second feature spaces, respectively, is 99% and the RMSE value for reconstruction error is approximately 0.0006, which is small compared to the average magnitude of the predicted quantities (0.01).

5.5.3 Surrogate Model Construction and Cross-Validation

After the important features are identified, several types of surrogate models are investigated as discussed in Section 5.2.2 and based on the cross-validation results given in Table 5.2, the Extra-Trees regressor is observed to give the best results with an average RMSE value of 0.10513 (15% better than the next best model). The LOOCV error is used to evaluate the surrogate model accuracy. The inputs to the surrogate model are TBC thickness, T1T rate, transient and constant creep rates and the outputs of the surrogate model are the top 20 important features.

Table 5.2: 7-fold cross-validation results in terms of RMSE

Fold #	Model					
	Extra-Trees	LightGBM	XGBoost	CatBoost	DNN	GP
1	0.1722	0.2029	0.1823	0.1780	0.2102	0.2205
2	0.1885	0.2146	0.2113	0.1843	0.2243	0.2315
3	0.1329	0.1829	0.1732	0.1362	0.1856	0.1795
4	0.1090	0.1257	0.1219	0.1155	0.1351	0.1583
5	0.1123	0.1286	0.1199	0.1136	0.1209	0.1348
6	0.1039	0.1273	0.1166	0.1065	0.1385	0.1421
7	0.1426	0.1835	0.1462	0.1503	0.1857	0.1812
Average	0.1373	0.1665	0.1531	0.1406	0.1714	0.1782

For illustration purposes, the predicted feature values versus the actual feature values are shown

in Fig. 5.5 for FEM run 23. Note that the pairs of observations and predictions are close to the 45-degree line, showing good agreement. The R^2 and RMSE values of the predictions shown in Fig. 5.5 are 0.97 and 0.45 (small error compared to the actual feature values), respectively. Figure 5.6 shows actual Feature 1 vs. the input parameters TBC thickness and T1T rate. Although the general trend shows a decrease of Feature 1 values with an increase in TBC thickness, there are still small Feature 1 values across all values of TBC thickness. Feature 1 values increase as T1T rate increases, indicating a possible quadratic relationship. From the physics of the problem, it is known that T1T values greater than 90 result in strong creep cases (i.e., large CEEQ and Dc values) and Feature 1 values corresponding to these cases are significantly larger than Feature 1 values corresponding to the weak creep cases (i.e., T1T < 90). Moreover, the correlations between the model inputs and the top 5 important features are computed (not shown in the chapter due to space limitation), and the most significant correlation is found to be between T1T rate and Feature 1, with a value of 0.85.

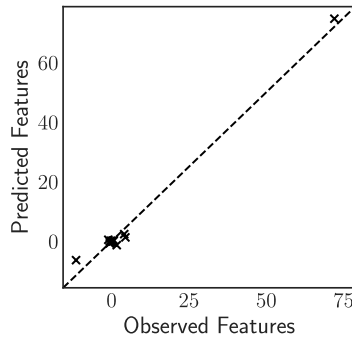


Figure 5.5: Predicted vs. observed values of important features for FEM run 23

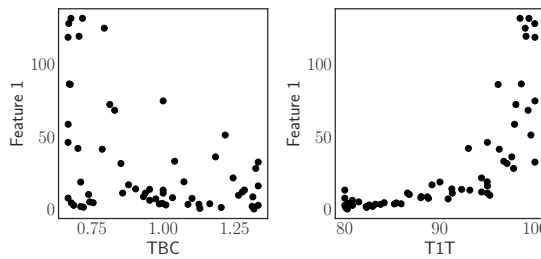


Figure 5.6: Actual Feature 1 values vs. TBC thickness and T1T rate

5.5.4 Adaptive Surrogate Modeling

Next, the training points corresponding to additional FEM runs (47-66) are adaptively selected based on the proposed methodology of adaptive training point selection. These runs are completed in 4 batches (see Fig. 5.7). The first batch is FEM 47-56, the second batch is FEM 57-60, the third

batch is FEM 61-62, and the fourth batch is FEM 63-66. From the surrogate model improvement point of view, it is more desirable to select one training point at a time. However, this is not possible in industrial settings due to the fact that running a single FEM model at each step is inefficient use of human and computational resources. A more efficient strategy in this application was to run between 2 and 10 FEM models at the same time. Thus, multiple new training points were proposed in each batch where the number of points for each batch is chosen based on error quantification, space filling, as well as expert opinion regarding inputs of interest. The simplest strategy would be to rank the candidate points and select the desired number of best points. However, this is not ideal since it does not consider the information overlapping of the new points in that batch, which can lead to clustered batch points [154]. (This is because the space-filling component of the learning function in Section 5.2.4.3 only gives the score for each single candidate point in terms of its distance from the existing points; whereas when we consider multiple points in a batch, we also have to consider the distances among them). In our case, we select the new training points in each batch by considering how informative and diverse they are [154], from the perspectives of both geometry and physical behavior. This way not only are the new points sampled in the regions of interest but they are also far away from each other. For example, in the second batch, among the 10 candidate points considered, only four (57-60) were selected that satisfied the criteria of informativeness and diversity. Similarly, in the third and fourth batches, only two points each were selected for the same reasons.

The first batch of new training points are selected by performing the proposed methodology described in Algorithm 3 on the previous 46 FEM runs, and the second batch of new training points are obtained by doing the same analysis on the previous 56 FEM runs (i.e., 46 initial points plus 10 points from the first batch of additional training points), and so on. For most of the adaptively sampled points, i.e., FEM 47-61, and 63, the parameters of the learning function are estimated by using the two-step algorithm described in Section 5.2.4.3. Most of these samples are from regions where the absolute error has high values (i.e., physical quantities take medium to high values, which is of interest to the analyst in this case). However, for the problem of interest it is also equally important to be able to accurately predict physical quantities that are close to zero. (For example, in the gas turbine engine, some input values might lead to low creep; such cases could be quickly screened and ignored in further detailed analysis). Thus, for a few of the adaptively selected additional samples such as FEM 62, 64, 65 and 66, the parameters of the learning function shown in Eq. (5.5) are chosen as unity to give equal weight to all metrics in order to facilitate choosing more diverse training points, including from regions where the relative error is higher, within each batch

(e.g., FEM 64 and 65).

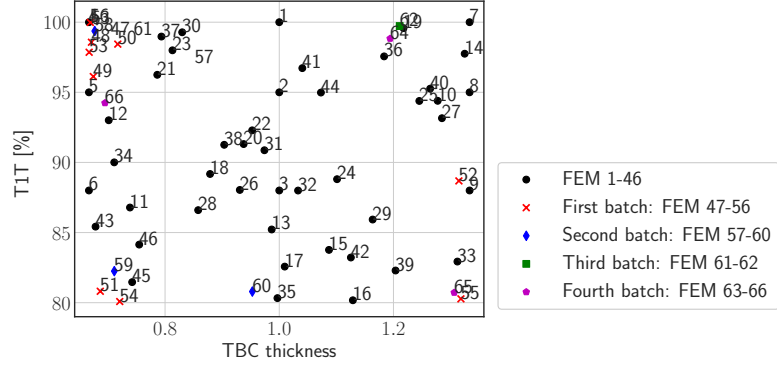


Figure 5.7: Initial FEM training data and adaptively selected additional samples

The surrogate model prediction accuracy is further analyzed in the original space by mapping the surrogate model predictions in the features space back to the original space. As discussed earlier, the percentage of nodes having $MAE > \text{threshold}$ (PNMAE) and the percentage of nodes having $rMAE > \text{unity}$ (PNrMAE) are defined for error analysis. The threshold value of 0.00025 is chosen based on expert opinion. The accuracy of the final surrogate model is shown in Fig. 5.8 using the two error metrics PNMAE and PNrMAE in a two-dimensional space, i.e., TBC thickness vs TIT rate. The error metrics have different values in different regions. For example, the absolute error metric, PNMAE, has smaller values in the bottom half region, where the physical quantities are close to zero, and has larger values in the top left region, where the physical quantities take medium to high values. Whereas, the relative error of the predicted creep response, PNrMAE, has greater values in regions where weak creep behavior is dominant. This is caused by dividing with a very small number. Since the physical quantities are close to zero in weak creep regions (i.e., the bottom half region $TIT < 90$) the denominator of Eq. (5.2) would be close to zero.

The final surrogate model predictions after 66 training points at an important node (id 5057: hot spot, based on analyst’s knowledge) are compared against the FEM results of run 36, where high creep damage is observed, as shown in Fig. 5.9. It is observed that the predictions are in good agreement with the FEM results at all time steps (similar trends in some cases if not actual values), which reflects the effectiveness of the proposed methodology.

The LOOCV results are shown in Table 5.3 using three different error metrics. In order to have a fair comparison, the improvement in the surrogate model accuracy is assessed on the same data set (i.e., first 46 and 56 FEM runs). The columns of Table 5.3 represent the average LOOCV error values on the first 46 and 56 FEM runs. In general, the accuracy of the surrogate model is observed

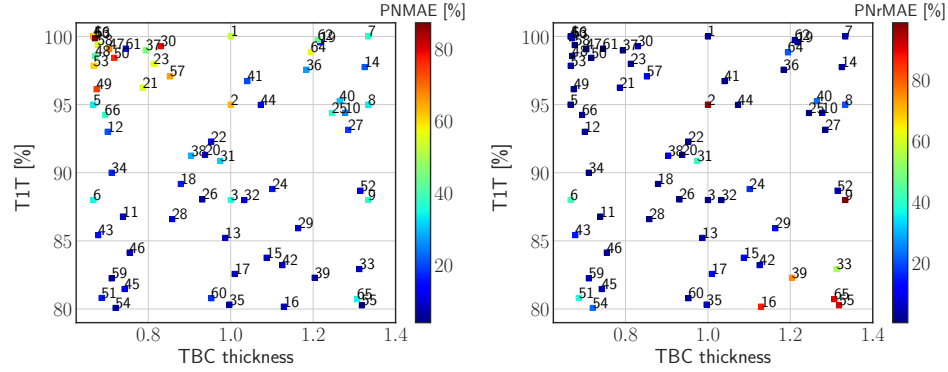


Figure 5.8: Final surrogate model prediction accuracy in the original space for leave-one-out FEM runs

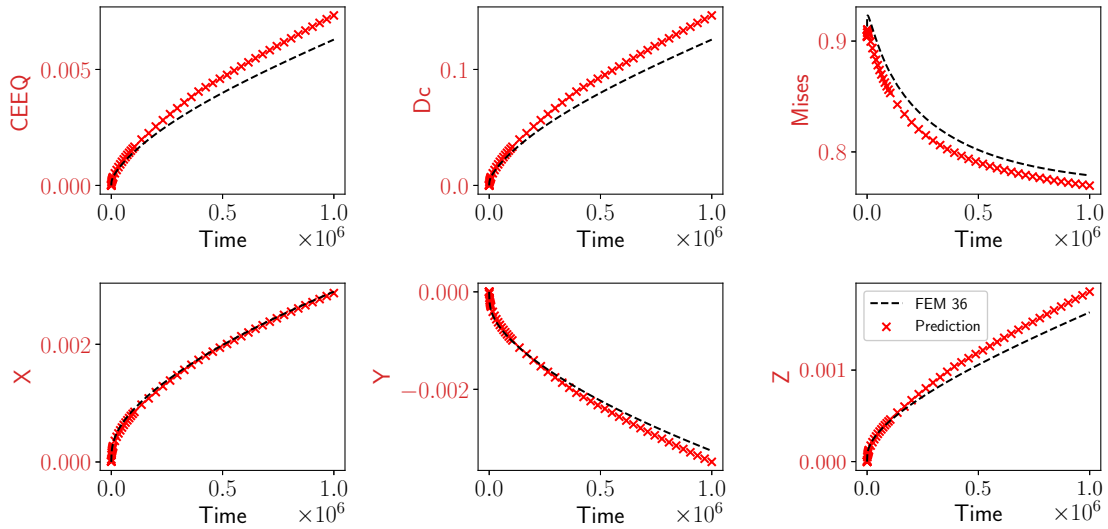


Figure 5.9: Final surrogate model prediction compared against FEM run 36 at node 5057

to improve with each batch of adaptively selected samples. The average RMSE value, which is obtained by averaging the RMSE values obtained for each LOOCV fold, decreases with each new batch of adaptively selected training points. For example, it decreases from 0.00254 to 0.00222 with the addition of 10 new adaptively selected training points (samples 47-56) based on the LOOCV on the first 46 FEM runs. Similarly, PNMAE and PNrMAE values decrease as we include adaptively selected training points during the surrogate model construction process.

The number of training points N and the surrogate model accuracy are used as stopping criteria. The sequential adaptive sampling procedure is stopped after 66 training points, $N = 66$. The adaptively improved surrogate model does not successfully reach the desired level of accuracy of

Table 5.3: Average leave-one-out cross-validation (LOOCV) results in PNMAE, PNrMAE, and RMSE

FEM runs considered	Avg. PNMAE of first 46 FEM runs	Avg. PNMAE of first 56 FEM runs	Avg. PNrMAE of first 46 FEM runs	Avg. PNrMAE of first 56 FEM runs	Avg. RMSE of first 46 FEM runs	Avg. RMSE of first 56 FEM runs
FEM 1-46	27.9	-	19.3	-	0.00254	-
FEM 1-56	26.	30.8	16.6	18.2	0.00222	0.00278
FEM 1-60	26.5	29.8	17.6	18.0	0.00223	0.00271
FEM 1-62	24.8	28.2	16.1	16.5	0.00206	0.00251
FEM 1-66	25.0	27.9	15.8	15.9	0.00195	0.00235

average PNrMAE $\leq 15\%$ (chosen based on expert opinion). However, the average PNrMAE value is 15.8% after 66 training points (for the first initial 46 FEM runs), which suggests that with a few more FEM runs the desired level of accuracy could be reached. In addition to the average PNrMAE value being distinctly close to the desired level of accuracy, the PNMAE and RMSE values are also acceptable for the analyst after 66 training points. All calculations except the FEM simulations are performed on a single desktop computer with an Intel 8-core CPU with a 3.00 GHz base frequency and 8 GB memory.

5.6 Summary

This chapter developed an approach for adaptive surrogate model construction in engineering problems with high-dimensional spatio-temporal output. The important features are obtained by performing a two-level dimension reduction using rSVD to map the original high-dimensional spatio-temporal output to an uncorrelated space. Cross-validation error is used to identify the most accurate surrogate model type. Once the most accurate surrogate model type is identified for the problem of interest, the prediction error in the original space is evaluated using LOOCV with different error metrics (i.e., PNMAE and PNrMAE). The subsequent adaptive sampling technique combines exploration and exploitation for adaptive improvement of surrogate model accuracy with the fewest possible runs of the expensive physics-based original model. The proposed adaptive sampling technique is compared with some of the existing adaptive schemes (Section 5.4). The proposed method yields very good results across all investigated test cases. The effectiveness of the proposed methodology is further demonstrated for a turbine blade example, using a time-dependent multi-physics dynamic system model with high-dimensional spatio-temporal outputs. It is observed that the adaptively improved surrogate model reaches an acceptable level of accuracy using the proposed strategy.

CHAPTER 6

Multi-Level Information Fusion for Model Calibration¹

6.1 Introduction

In most cases, the model parameters are not well-known, and model predictions may not agree with observational data. Thus, model calibration needs to be performed to determine the values of unknown parameters by requiring the model outputs to match experimental data. Several model calibration techniques are available in the literature (e.g., least squares, maximum likelihood estimation, and Bayesian estimation). The output of any model is affected by various sources of uncertainty. Bayesian methods provide a convenient framework for combining prior beliefs about parameters with evidence gained from data [48, 49]. For highly coupled multi-physics systems with sparse data and many model parameters, the heterogeneous sources of uncertainty and their relationships need to be organized in a systematic manner to facilitate effective use of available data. Bayesian networks (BNs) provide such a systematic approach, and represent the relationships between model inputs and outputs, uncertainty sources, and data [18, 19]. BNs can facilitate analyses in both forward and inverse directions. The uncertainty in the outputs can be estimated using forward propagation through the BN, given the uncertainty about various inputs, model parameters, and model errors. In the inverse problem, unmeasured inputs, system states, or model parameters can be estimated using Bayesian inference, given observations of some of the inputs and outputs. In general, they enable the inference of unmeasured quantities in a system model by using the observed data on measured quantities.

Uncertainty quantification (UQ) seeks to quantify and reduce the uncertainty arising from multiple, heterogeneous sources. UQ has two directions: the forward problem and the inverse problem. In the forward problem, model errors and the uncertainty related to the model inputs and parameters are propagated to compute the uncertainty in the output. In the inverse problem, model calibration (i.e., estimation of uncertain model parameters and errors) and system diagnosis (i.e., estimation of uncertain system states and errors) are performed using the available measurement data. The inverse problem is an essential part of uncertainty quantification and reduction and in achieving the desired level of prediction confidence.

In Bayesian calibration, prior and posterior beliefs about the calibration quantities can be rep-

¹Adapted with permission from: Kapusuzoglu, B., Mahadevan, S., Matsumoto, S., Miyagi, Y., & Watanabe, D., “Multi-Source Information Fusion for Multi-Component Dynamic System using Multi-Level Bayesian Calibration,” (under review), 2022.

resented as probability density functions [155]. Several studies have addressed the use of Bayesian calibration frameworks for UQ and prior selection in model calibration [50, 51]. Li and Mahadevan [156] used Bayesian inference to quantify the uncertainty from the lower-level models to the prediction of the system level. Mullins and Mahadevan [157] address the role of calibration and validation in multi-level uncertainty integration for hierarchical systems. There are several recent studies using Bayesian networks (BNs) [156–158] for model calibration. Sankararaman and Mahadevan [159] proposed a methodology to integrate model verification, validation, and calibration using BNs. They have extended the application of the method to the system with multi-level models sharing common calibration parameters. A BN was proposed by Hu and Mahadevan [160] to aggregate heterogeneous sources of information and to quantify the uncertainty in the quantity of interest (QoI). Nannapaneni and Mahadevan [158] also used a Hierarchical Bayesian network (HBN) as a mathematical framework for the aggregation of uncertainty from multiple sources to quantify the uncertainty.

Engineering analyses are often done in the presence of uncertainty due to the insufficient quality and quantity of available data. The information is typically sparse in realistic systems, and data might be collected at different time instances for different system components. When multiple models are used to describe system behavior (either multi-physics or multi-level), some unknown parameters are shared across multiple component models whereas some other unknown parameters are local to a given component. Also, some physical quantities might be measured for a single-component and some other quantities for multiple components. In addition, the different measurements might be available at different time instants, not all at the same instants. The calibration of model parameters in situations where the models have complicated couplings between them and are based on different physics is not straightforward. Simultaneous Bayesian calibration of all the model parameters can be computationally prohibitive. The existing research has focused mostly on the analysis of physical systems with single-level data, not involving multi-level heterogeneous data. A segmented approach for model calibration has been studied earlier for multi-physics problems [18, 19]. In this work, the segmented approach is extended to multi-level transient systems, in the presence of sparse experimental data for different system components at different time instants.

This chapter develops an approach that fuses multi-source information for offline and online Bayesian calibration of multi-physics models with multi-level data. Specific contributions are made in the following steps: (1) A BN is constructed in a hierarchical manner from component-level to system-level. (2) A particle filter (PF) algorithm, which is commonly used for state estimation in dynamic systems, is adopted in this paper for Bayesian inference of static model parameters. (3) Multi-level Bayesian calibration of model parameters is performed by integrating experimental data

for different components at different time instants. The calibration is pursued in two directions: offline and online. In the offline strategy, the calibration is performed using data that is collected over multiple time steps. In the online strategy, the calibration is performed in real-time as new measurements are obtained at each time step, thus continuously updating the model. Expert knowledge and physical measurements at different levels of the network enable effective calibration even with insufficient data.

6.2 Proposed Methodology

In many practical engineering systems accurately estimating the remaining useful life and real-time system response has been a challenge due to the complexity and uncertainty in service environments and multidisciplinary mechanisms. Several computationally expensive simulation model (FEM) runs with varying model parameters needs to be performed to quantify the effect of multiple uncertainty sources on the QoI. Finite element analysis of a large structure provides many output quantities (e.g., stresses and deformations along several degrees of freedom) at numerous locations. Surrogate models need to be constructed to replace these computationally expensive simulation models. When the output of the physics-based model is high-dimensional and spatio-temporal as in the problem of interest, the direct use of all the data simultaneously for Bayesian calibration is not practical.

The proposed methodology aims to develop an efficient Bayesian calibration approach in time-dependent systems through multi-source information fusion. Multiple types of measurement data from multiple components at different time instants are considered. A HBN approach is used to integrate expert knowledge and measurements from multiple domains at different levels of the network. Two strategies are investigated: offline calibration and online calibration. The information fusion at different levels is presented for both strategies and the differences are highlighted.

6.2.1 Bayesian Calibration with Multiple Types of Data

The measurement data \mathcal{D}_{obs} , for Bayesian calibration can be available for different components at different time instants and spatial locations in a multi-component system. Some of the model parameters can be local ψ_{local} (i.e., a unique value for each component) and others can be global ψ_{global} (i.e., a common value across components). Simultaneous calibration of all unknown parameters Θ can be computationally prohibitive with multi-component, asynchronous data without the use of a modular Bayesian calibration approach. Therefore an efficient methodology is proposed here to perform multi-level Bayesian calibration by fusing different data types from multiple components.

The proposed multi-level calibration approach leverages a hierarchical Bayesian network (HBN) where some nodes in a BN may represent another lower-level BN as shown in Fig. 6.1. Some of the nodes in the lower-level BN can also represent further lower-level BNs, thus any number of levels are possible. The inverse problem of Bayesian calibration is achieved by passing the information upstream from the data nodes (solid squares) to the calibration quantities. The data in each lower-level BN (i.e., $d_1^{(1)}$ and $d_2^{(1)}$) is used to estimate the posterior distributions of the nodes in the lower-level BN (e.g., $R_1^{(1)}, R_2^{(1)}, \dots, R_7^{(1)}$). During calibration of the nodes in the higher-level BNs, the posterior distributions of the lower-level nodes are used as prior distributions to re-calibrate the nodes that go into higher-level BNs (i.e., $R_1^{(2)}$ and $R_2^{(2)}$). If necessary, the remaining nodes in the lower-level BNs can also be re-calibrated by passing the information down the hierarchy. Thus, we are able to incorporate all the available information into the calibration strategy and maximize the computational efficiency.

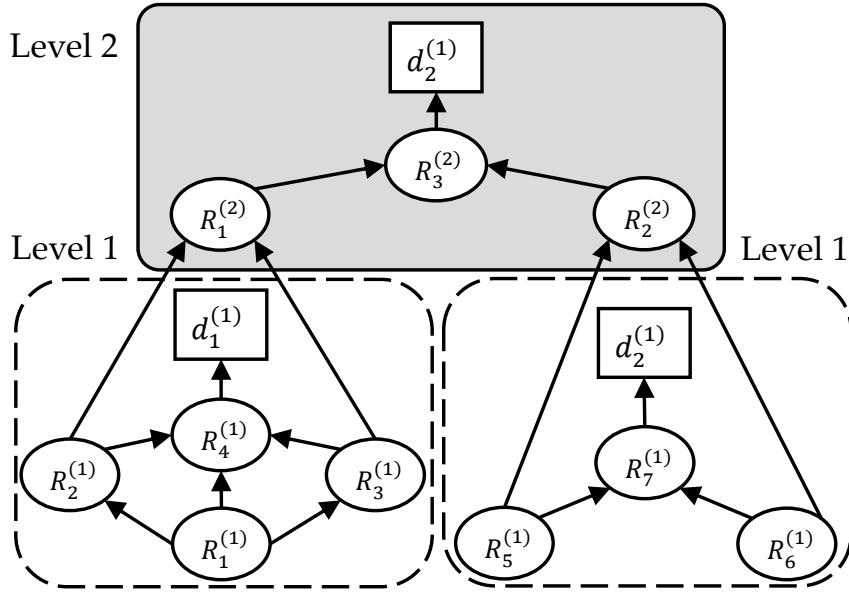


Figure 6.1: A simple hierarchical Bayesian network

The above development of the approach is under a static Bayesian framework. Extension to a dynamic BN with time-dependent system parameters is given in Fig. 6.2. The subscripts $t - 1$ and t denote the time instants and the nodes in the dynamic Bayesian network (DBN) are connected by arrows that represent conditional probability distributions or deterministic functional relations. In addition to the dynamic nodes for which the states change over time (e.g., node B, C, D, E), there can also be static nodes that are included at all time instances (e.g., node A). The situation considered in the numerical example in Section 5.5 is simpler. Since the numerical example does not

include any time-dependent uncertain system states, there is no calibration of the dynamic system states using the PF algorithm described in Section 2.3.2; only the estimation of static parameters.

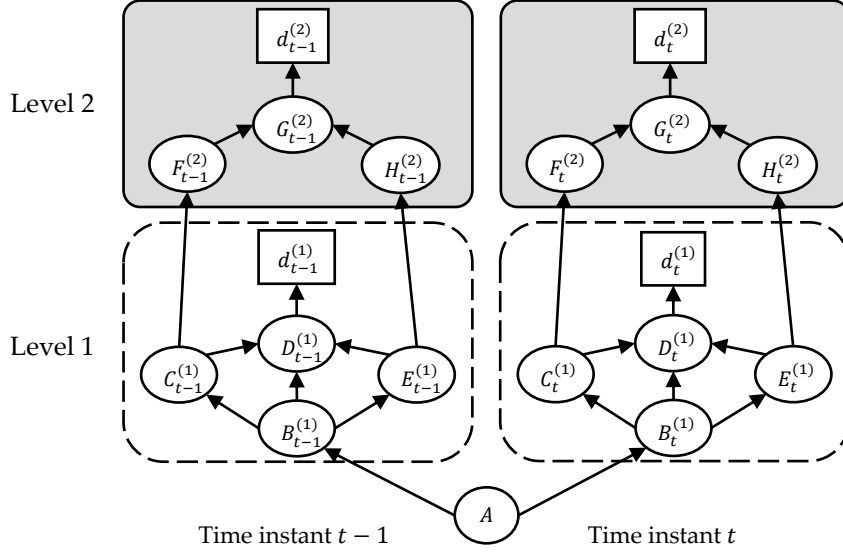


Figure 6.2: A simple hierarchical dynamic Bayesian network

Given all unknown parameters Θ , let Θ_1 and Θ_2 denote the calibration parameters, and let \mathcal{D}_{BN_1} and \mathcal{D}_{BN_2} represent the data available on the lower- and higher-level BNs, respectively. We consider two cases: (1) the available data \mathcal{D}_{BN_1} and \mathcal{D}_{BN_2} are independent, or (2) there is a one-to-one correspondence between them. Assuming \mathcal{D}_{BN_1} and \mathcal{D}_{BN_2} are independent, the posterior distributions $\Pi(\Theta_1, \Theta_2 | \mathcal{D}_{BN_1}, \mathcal{D}_{BN_2})$ can be obtained as

$$\begin{aligned}
 \Pi(\Theta_1, \Theta_2 | \mathcal{D}_{BN_1}, \mathcal{D}_{BN_2}) &\propto L(\mathcal{D}_{BN_1}, \mathcal{D}_{BN_2} | \Theta_1, \Theta_2) \Pi(\Theta_1, \Theta_2) \\
 &\propto L(\mathcal{D}_{BN_1} | \Theta_1) L(\mathcal{D}_{BN_2} | \mathcal{D}_{BN_1}, \Theta_1, \Theta_2) \Pi(\Theta_1) \Pi(\Theta_2) \\
 &\propto L(\mathcal{D}_{BN_1} | \Theta_1) \Pi(\Theta_1) L(\mathcal{D}_{BN_2} | \Theta_1, \Theta_2) \Pi(\Theta_2)
 \end{aligned} \tag{6.1}$$

where the first two terms in the last expression denote the posterior distributions of Θ_1 , $\Pi(\Theta_1 | \mathcal{D}_{BN_1})$, using the lower-level data \mathcal{D}_{BN_1} . The posterior distributions of Θ_1 are then used as prior distributions to obtain the posterior distributions of Θ_1 and Θ_2 using the higher-level data \mathcal{D}_{BN_2} . Since \mathcal{D}_{BN_1} and \mathcal{D}_{BN_2} are assumed to be independent, the likelihood term $L(\mathcal{D}_{BN_2} | \mathcal{D}_{BN_1}, \Theta_1, \Theta_2)$ simplifies to $L(\mathcal{D}_{BN_2} | \Theta_1, \Theta_2)$. In other words, the posterior distributions obtained in the lower-level BNs using data in that level are propagated as priors to the higher-level BN for re-calibration using the higher-level data \mathcal{D}_{BN_2} .

In the second case, the data \mathcal{D}_{BN_1} and \mathcal{D}_{BN_2} are assumed to have a one-to-one correspondence

(i.e., there exist a corresponding \mathcal{D}_{BN_1} for a given \mathcal{D}_{BN_2}), then the posterior distribution is given as

$$\begin{aligned}
\Pi(\Theta_1, \Theta_2 | \mathcal{D}_{BN_1}, \mathcal{D}_{BN_2}) &\propto L(\mathcal{D}_{BN_1}, \mathcal{D}_{BN_2} | \Theta_1, \Theta_2) \Pi(\Theta_1, \Theta_2) \\
&\propto L(\mathcal{D}_{BN_1} | \Theta_1) L(\mathcal{D}_{BN_2} | \mathcal{D}_{BN_1}, \Theta_1, \Theta_2) \Pi(\Theta_1) \Pi(\Theta_2) \\
&\propto L(\mathcal{D}_{BN_1} | \Theta_1) \Pi(\Theta_1) L(\mathcal{D}_{BN_2} | \mathcal{D}_{BN_1}, \Theta_2) \Pi(\Theta_2)
\end{aligned} \tag{6.2}$$

where the likelihood term $L(\mathcal{D}_{BN_2} | \mathcal{D}_{BN_1}, \Theta_1, \Theta_2)$ simplifies to $L(\mathcal{D}_{BN_2} | \mathcal{D}_{BN_1}, \Theta_2)$ since $R_3^{(2)}$ is independent of Θ_1 when \mathcal{D}_{BN_1} is known. The first two terms provide the posterior distributions of Θ_1 using \mathcal{D}_{BN_1} similar to the previous case, and the last two terms provide the posterior distributions of Θ_2 using \mathcal{D}_{BN_1} and \mathcal{D}_{BN_2} . Thus, the final posterior distributions of Θ_1 , Θ_2 can be obtained in one shot separately, whereas a two-step approach is used in the earlier case described in Eq. (6.2.1) to obtain the posterior of Θ_1 : Step 1: Calibrate using only \mathcal{D}_{BN_1} ; Step 2: Use the posterior from step 1 to re-calibrate together with Θ_2 using \mathcal{D}_{BN_2} . The above approaches can be extended to calibrate hierarchical DBNs with multiple levels.

The segmented calibration strategy discussed in [19] uses a static BN with multiple sources of calibration data. However, this strategy is only effective when data are available for each calibration segment. Since in multi-component dynamic systems one data source is often used to calibrate multiple sources of uncertainty, the strategies investigated here allow to quantify model error and calibration quantities within multi-level transient systems.

6.2.2 Offline and Online Multi-Level Bayesian Calibration

In this work, the multi-level Bayesian calibration is pursued in two directions: Offline and Online (see Algorithm 4). In the former strategy, the calibration of all parameters is performed at once using all the data, with data on different components that are collected at different time steps. In the latter strategy, the calibration is performed in real-time as measurements are obtained at each time step. In the offline strategy, Ψ_{global} are fixed at their priors and Ψ_{local} are calibrated for each system component using measurements at each time step and the posterior distributions from this calibration are then used as prior distributions to calibrate the global parameters while fixing the non-global parameters at their posterior estimates from the previous iteration step. Whereas in the online strategy, the same procedure is performed only at the time step where measurements are taken. The posteriors obtained from this data type are propagated in time and used as priors to further update the parameter estimates using additional data. As new measurements are taken, the real-time calibration is performed and unlike the offline strategy different data types are used

sequentially to update the parameters at the current time step in the online calibration strategy.

The calibration strategy as described in Algorithm 4 consists of the following general steps:

1. First assume priors for the parameters to be calibrated (both local and global parameters);
2. Next, data collected from multiple components is used to calibrate both ψ_{local} and ψ_{global} in an iterative manner until convergence;
3. More specifically, first the global parameters are fixed at their priors and the local parameters are calibrated for each sample;
4. Then, the local parameters are fixed at their posterior distributions to calibrate ψ_{global} ;
5. Steps 3 and 4 are then repeated until convergence, i.e., until the change in the mean and variance of the posterior distribution of ψ_{global} is smaller than a set threshold. (This is similar to the well-known expectation maximization (EM) algorithm [150]);
6. The posterior distributions obtained in step 5 are used as priors for the calibration of ψ_{local} corresponding to the component, where data is available.

The main difference between the offline and online strategies is that the offline strategy has access to complete measurement history. This allows the use of all available multi-component data for calibration (lines 2-8 in Algorithm 3). Whereas, in the online strategy the multi-component data is fed into the calibration strategy in real-time (lines 15-20 in Algorithm 3). This can lead to a different convergence behavior between the offline and online strategies since the iterative procedure performed to calibrate the parameters using multi-component data is dependent on the data that is fed into the calibration strategy.

The proposed methodology consists of five main components: (1) Collection of simulation data and dimension reduction, (2) Surrogate model construction for fast system output prediction, (3) Collection of experimental data, (4) Information fusion to calibrate the unknown parameters and update the surrogate model with real-time monitoring data and model discrepancy, and (5) Prediction of the real-time QoIs using the calibrated system model. In the next section, we demonstrate the effectiveness of the proposed methodology for a high-dimensional thermo-mechanical system with spatio-temporal output.

Algorithm 4 Multi-level Bayesian calibration pseudo code:

(A) Offline calibration, and (B) Online calibration

Given: \mathcal{D}_{obs}

Output: Posterior estimates of Θ

```

1: procedure (a) Offline Calibration
2:   for  $i = 1$  to  $n_{time}$  do
3:     while (Change in posteriors of  $\Psi_{global} > \text{threshold}$ ) do
4:       Fix  $\Psi_{global}$  at their priors
5:       Calibrate  $\Psi_{local}$ 
6:       Fix  $\Psi_{local}$  at their posterior estimates and calibrate  $\Psi_{global}$ 
7:     end while
8:   end for
9:   for  $i = 1$  to  $n_{time}$  do
10:    Re-calibrate both  $\Psi_{local}$  and  $\Psi_{global}$ 
11:  end for
12: end procedure
13:
14: procedure (b) Online Calibration
15:   for  $i = 1$  to  $n_{time}$  do
16:     while (Change in posteriors of  $\theta_{global} > \text{threshold}$ ) do
17:       Fix  $\Psi_{global}$  at their priors
18:       Calibrate  $\Psi_{local}$ 
19:       Fix  $\Psi_{local}$  at their posterior estimates and calibrate  $\Psi_{global}$ 
20:     end while
21:     Re-calibrate both  $\Psi_{local}$  and  $\Psi_{global}$ 
22:   end for
23: end procedure

```

6.3 Numerical Example

6.3.1 Introduction to the Problem

A simplified gas turbine engine blade model is studied in this section to demonstrate the proposed surrogate modeling and calibration techniques for high-dimensional spatio-temporal output. The field response outputs (i.e., quantities of interest, QoIs) of the FEM runs are equivalent creep strain (CEEQ), creep damage (Dc), von Mises stress, displacement in x, y, and z directions; these are available at a large number of spatial and temporal points. The four-dimensional input for the surrogate model consists of turbine blade coating thickness (TBC thickness), turbine output rate (T1T rate), transient creep rate, and constant creep rate, as shown in Table 6.1. The ranges of the inputs are: TBC thickness $[2/3 \cdot \text{nominal}, 4/3 \cdot \text{nominal}]$, T1T rate $[80 \cdot \text{nominal}, 100 \cdot \text{nominal}]$, and transient and constant creep rates $[-3\sigma, 3\sigma]$. The boundary conditions of the FEM model are assumed to be known with certainty. The four surrogate model inputs (i.e., calibration quantities) are considered as uncertain. The total number of spatial locations of the FEM model is over 27,000 for each output QoI and the total number of time steps is 54. Thus, for each QoI, a single FEM

simulation gives a vector with over 1,458,000 spatio-temporal quantities.

Table 6.1: Surrogate model input parameters and ranges

Input	Values
TBC thickness	$\frac{2}{3} \times \text{nominal}$ to $\frac{4}{3} \times \text{nominal}$
T1T rate	$80 \times \text{nominal}$ to $100 \times \text{nominal}$
Transient creep rate	-3σ to 3σ
Constant creep rate	-3σ to 3σ

6.3.2 Bayesian Parameter Calibration

We use the particle filter (PF) algorithm to compute the posterior distributions of $\mathbf{X} = [\text{TBC thickness, T1T rate, } \alpha, \gamma]$ as shown in Fig. 6.3. A uniform distribution is chosen as the prior distribution for each parameter of \mathbf{X} .

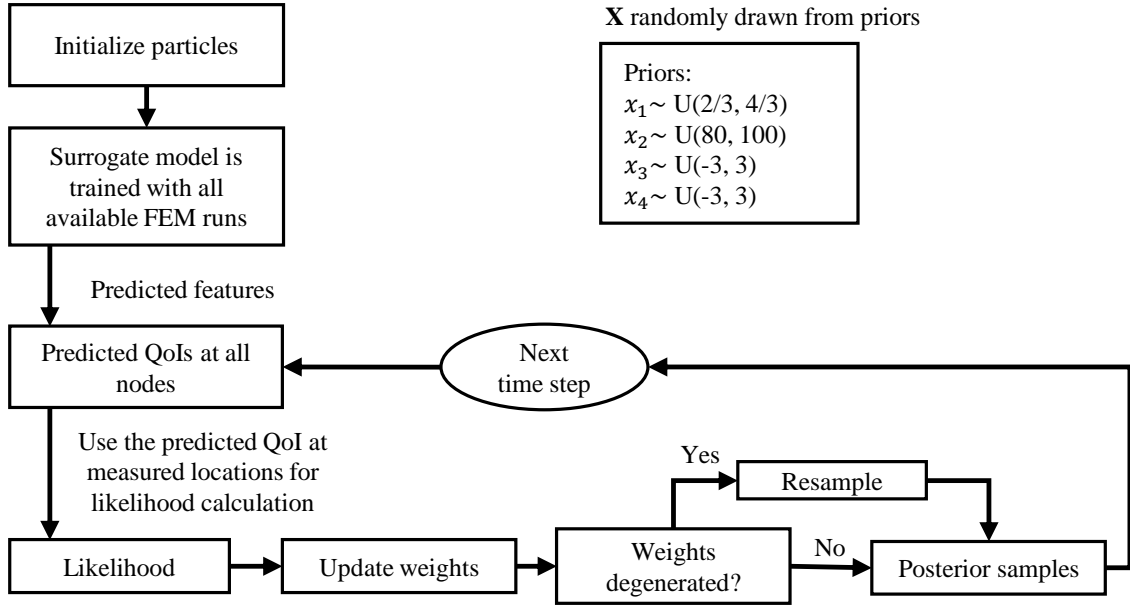


Figure 6.3: Flowchart describing the steps of particle filter algorithm

The developed surrogate model in Chapter 5 through adaptive selection of additional training points is used in this work to calibrate the model parameters. In sequential updating using the PF method, for each proposed \mathbf{X} , the important features are calculated using the corresponding surrogate model that is trained with 66 FEM runs. The physical quantities in the original space are subsequently calculated by performing an inverse projection of the important features onto the original space. The likelihood function shown in Eq. (2.20) is calculated while considering a

stochastic observed measurement error $\boldsymbol{\varepsilon}_{obs}$ that is represented as a zero-mean Gaussian random variable with an unknown variance σ_{obs}^2 , i.e., $\boldsymbol{\varepsilon}_{obs} \sim N(0, \sigma_{obs}^2)$.

There are over 27,000 spatial locations for each QoI (i.e., FEM nodes) and there are total of 54 time steps. However, the measurement data is significantly sparse. We perform multi-level Bayesian calibration using different types of measurement data as described in Fig. 6.4 to make use of all the data. We have single-blade (single-component) data and multi-blade (multi-component) data. Elongation at the blade tip is available for 100 blades, and x, y, z displacement data is only available for a single blade (i.e., blade ID 01). Both elongation at the blade tip and x, y, z displacement data are collected at the same three time instants. These data types have measurements at three time instances. For example, the elongation data at the blade tip (i.e., Node ID 31933) is collected from 100 blades and only at three time instants. The 3D deformation data is obtained at the same three time instants at 6448 different spatial locations for blade ID 01. We have CEEQ data only for three blades. CEEQ measurements are obtained by destructive testing, which is different from collecting displacement data with a laser scanner (non-destructive). After the destructive testing of a blade, another blade will be installed as replacement. Thus, CEEQ data for a single blade is only available at one time instant and at a single spatial location (i.e., Node ID 5057) as shown in Table 6.2. Therefore, the amount of data coming from CEEQ measurements is even more limited, CEEQ values at a single spatial location for three time steps, (i.e., three measurement values for each blade compared to $\approx 27,000 \times 54$ surrogate model predictions). Furthermore, the two creep parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$ are not strongly correlated to the QoIs. Thus the uncertainty regarding these parameters is not reduced with the limited available data. Because of all these reasons, the posterior estimates obtained using CEEQ measurements are not shown since the available CEEQ measurements are very sparse (i.e., the uncertainty regarding local parameters is not reduced significantly).

Table 6.2: Experimental data

Data type	Time instants		
	t_1	t_2	t_3
Elongation at the blade tip (Node ID 31933)	blade ID 01-100	blade ID 01-100	blade ID 01-100
XYZ (at 6448 different spatial locations)	blade ID 01	blade ID 01	blade ID 01
CEEQ (Node ID 5057)	blade ID 28	blade ID 82	blade ID 99

The multi-level Bayesian calibration approach is implemented along the following steps: (1) Assume priors for the four parameters to be calibrated, where three of them $\Theta_{local} = \{\text{TBC thickness}, \boldsymbol{\alpha}, \boldsymbol{\gamma}\}$ are local parameters, i.e., unique to each blade, and $\Theta_{global} = \{\text{T1T}\}$ is a global parameter across all blades. (2) Use multi-blade data, i.e., tip elongation of 100 blades to calibrate Θ_{local} and Θ_{global} in an iterative manner as explained in Section 6.2.2 and shown in Fig. 6.4(b). (3) Use

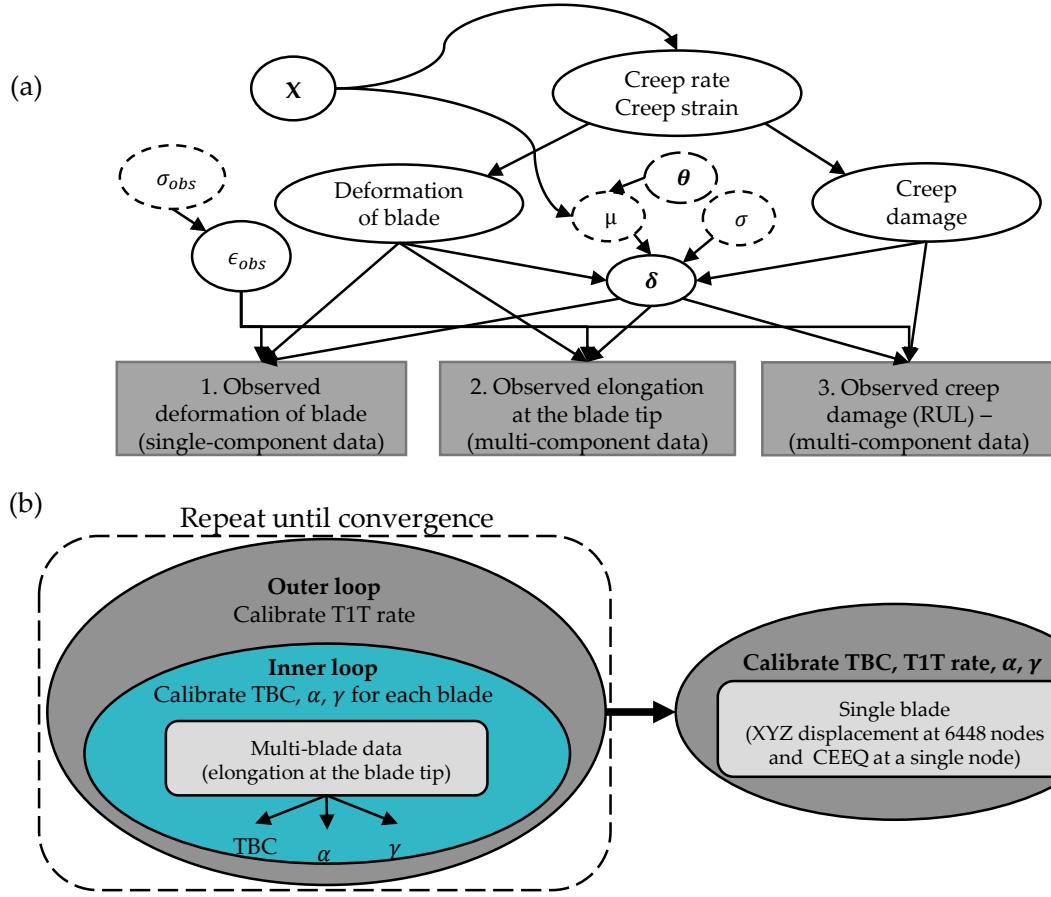


Figure 6.4: (a) Multi-level Bayesian network and (b) Offline calibration in multi-component system

the posterior distributions obtained in the above step as priors to re-calibrate Θ_{local} (3 parameters that are local to blade ID 01) using the single-blade 3D deformation data of blade ID 01 shown in Fig. 6.4(b). (4) Use the posterior distributions of Θ_{local} and Θ_{global} obtained from the second step (using the multi-blade data) as priors for re-calibrating all parameters using CEEQ measurements of blade ID 28, 82, and 99 at three unique time instances (see Table 6.2).

Two options of multi-level Bayesian calibration are pursued: offline and online. In the former strategy, the calibration is performed all at once, using all the data that is collected at multiple time steps. In the latter strategy, the calibration is repeated in real-time as measurements are obtained at each time step. The resulting posterior distributions of parameters based on the offline Bayesian calibration corresponding to blade ID 90 obtained by using the multi-blade measurement field data (i.e., elongation at the blade tip) are shown in Figs. 6.5 and 6.6. The posterior distributions of parameters corresponding to blade ID 01 obtained by using the 3D deformation data based on the offline Bayesian calibration are given in Figs. 6.7 and 6.8.

The true values of TBC thickness, α and γ are unknown, and the true value of T1T rate is not exactly known but based on expert opinion it is expected to be in the range $[0.97, 1.0]$, respectively, for all samples (since it is a shared parameter). The results with an input-output dependent model discrepancy term ($\delta \sim N(\mu(x,y,\theta), \sigma(x,y,\theta))$, with $\mu(x,y,\theta) = y(\theta_1 - \theta_2x)$, see Section 2.3.1) are plotted together with results that do not consider model discrepancy (labeled as Posterior without δ) but only consider measurement error ϵ_{obs} . In the presence of insufficient amount of experimental data and non-informative prior knowledge about the uncertainty sources in the engineering system, it may be difficult to distinguish between the effects of the model parameters and model discrepancy as the number of parameters that need to be estimated becomes large; this problem is referred to as non-identifiability [51, 161]. Hence, we assume a constant value for the standard deviation of model discrepancy (i.e., $\sigma(x,y,\theta) = \sigma = 1 \times 10^{-5}$) based on the analysis of surrogate model predictions and measurement data. The inclusion of model discrepancy allows to have slightly sharper posterior distributions for the calibration parameters as additional single-component data is used re-calibration. This can be seen in Figs. 6.7, 6.8.

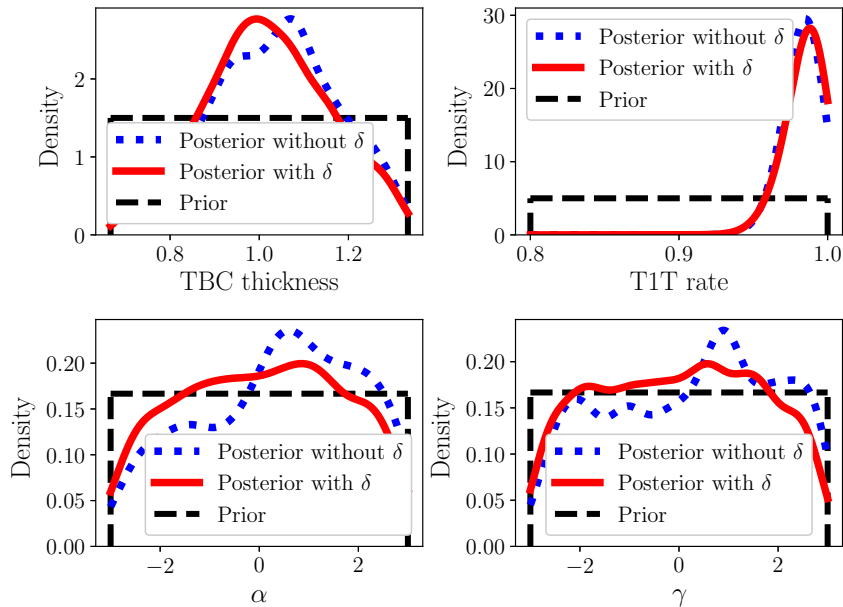


Figure 6.5: Posterior distributions of model parameters using elongation data at the blade tip (blade ID 90), using offline Bayesian calibration

The calibrated prediction is verified with measurement data. Some of the measurements are not used to calibrate the parameters. Hence, the calibrated system model predictions are verified at these unused measurement data. The mean values and standard deviations of the calibrated parameters, which are obtained from the weights of particles representing samples from the joint

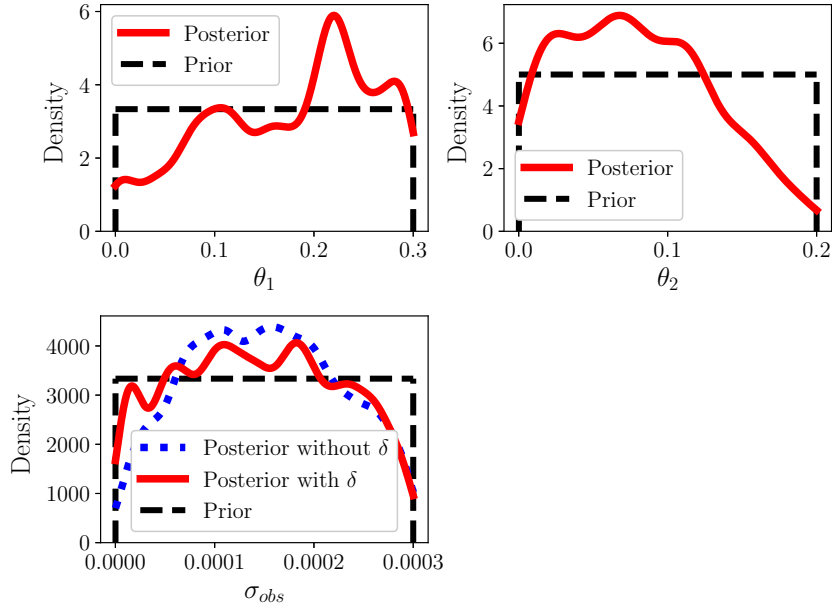


Figure 6.6: Posterior distributions of noise standard deviation and model discrepancy hyperparameters using elongation data at the blade tip (blade ID 90), using offline Bayesian calibration

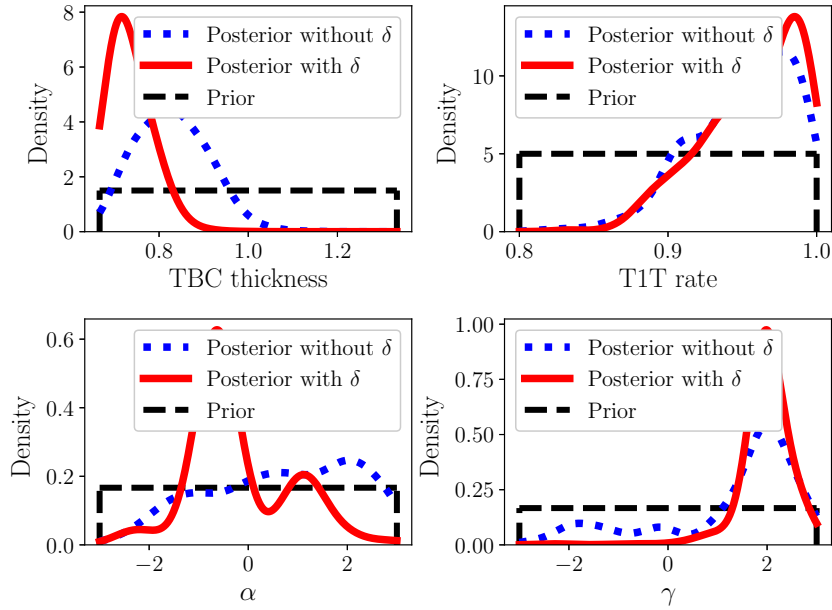


Figure 6.7: Posterior distributions of model parameters using 3D deformation data of blade ID 01, using offline Bayesian calibration

distribution, are used to propagate the uncertainty through the surrogate model. The displacement predictions obtained using the calibrated parameters are compared against the measurements at Node ID 6448 to validate the calibration strategy.

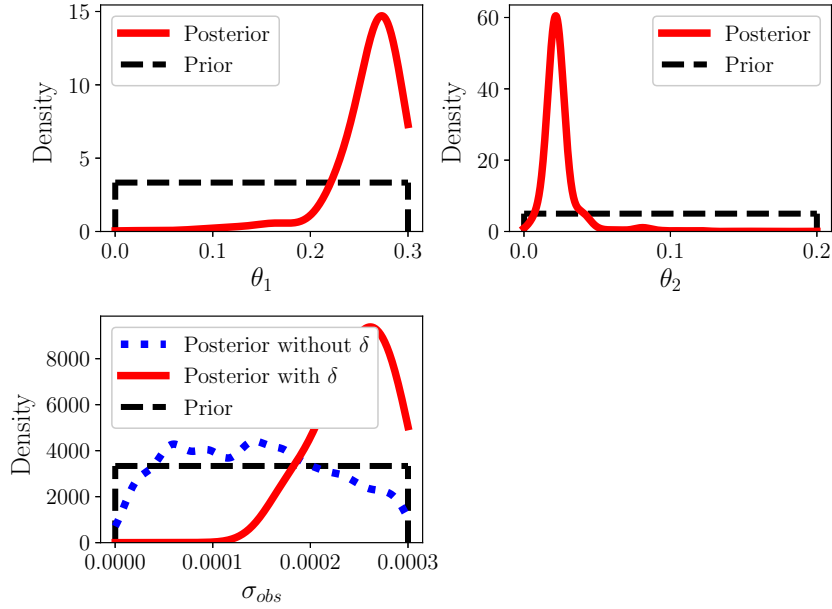


Figure 6.8: Posterior distributions of noise standard deviation and model discrepancy hyperparameters based on 3D deformation data of blade ID 01, using offline Bayesian calibration

The calibrated surrogate model predictions of the offline Bayesian calibration are compared against the measurement data in Figs. 6.9 and 6.10. The blue squares denote the mean surrogate model predictions. The light blue shaded region represents the surrogate model predictions at the mean plus or minus one standard deviation of the calibrated parameters ($\mu \pm \sigma$). Similarly, the red error bar indicates the surrogate model predictions at the mean plus or minus one standard deviation of the calibrated parameters plus the measurement error ($\mu \pm \sigma + \varepsilon_{obs}$) in Figs. 6.9(a) and 6.9(b) and model discrepancy ($\mu \pm \sigma + \varepsilon_{obs}$) in Figs. 6.10(a) and 6.10(b). The measurements at each time step are within the prediction bounds obtained using the posterior estimates of the calibrated parameters, demonstrating the effectiveness of the proposed method.

The posterior distributions of parameters based on the online Bayesian calibration are shown in Figs. 6.11, 6.12, 6.13, and 6.14 for calibration with different types of measurement data. The calibrated surrogate model predictions are compared against the measurement data as shown in Figs. 6.15 and 6.16 to assess the performance of the proposed online strategy.

The offline strategy, in which data at all time steps are used at once to calibrate the model parameters, is able to capture slightly more information. Thus, when we compare Figs. 6.7 and 6.8 with Figs. 6.13 and 6.14, we observe that the posterior distributions obtained using the offline strategy significantly sharper. Also note that the single-blade data type has measurements available

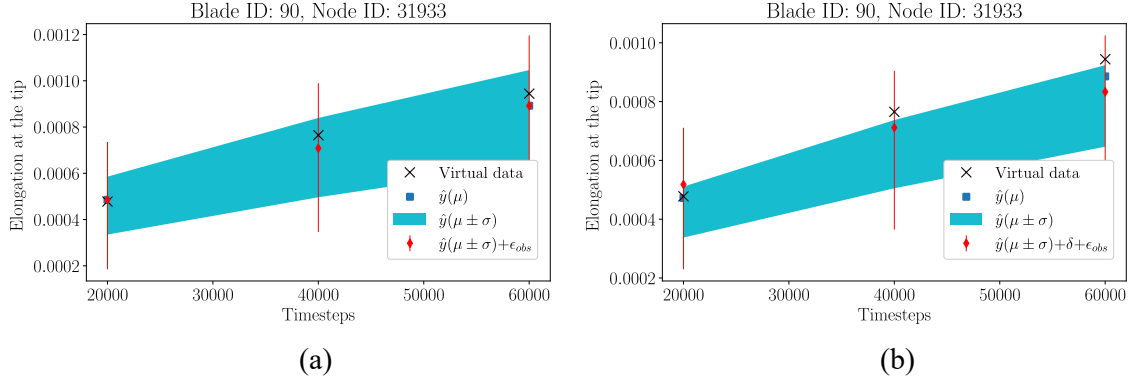


Figure 6.9: Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the elongation at the blade tip data of blade ID 90, using offline Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy

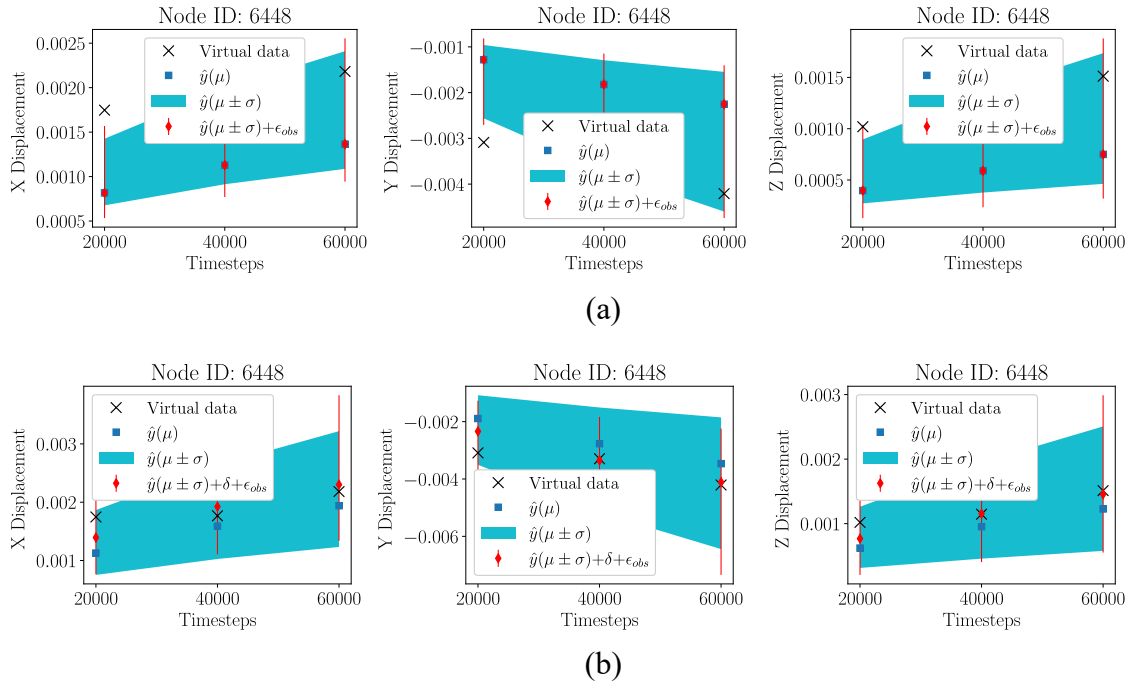


Figure 6.10: Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the 3D deformation data of blade ID 01, using offline Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy

only for one blade, thus the information gain is smaller compared to the multi-blade data type where there are multiple pieces of data. Based on the results, it seems more reasonable to perform the offline calibration strategy to capture as much information as possible in the first level calibration (i.e., iterative multi-component calibration), so that the posterior distributions of local parameters would be sharper after the second level calibration (i.e., single-component calibration such as: CEEQ

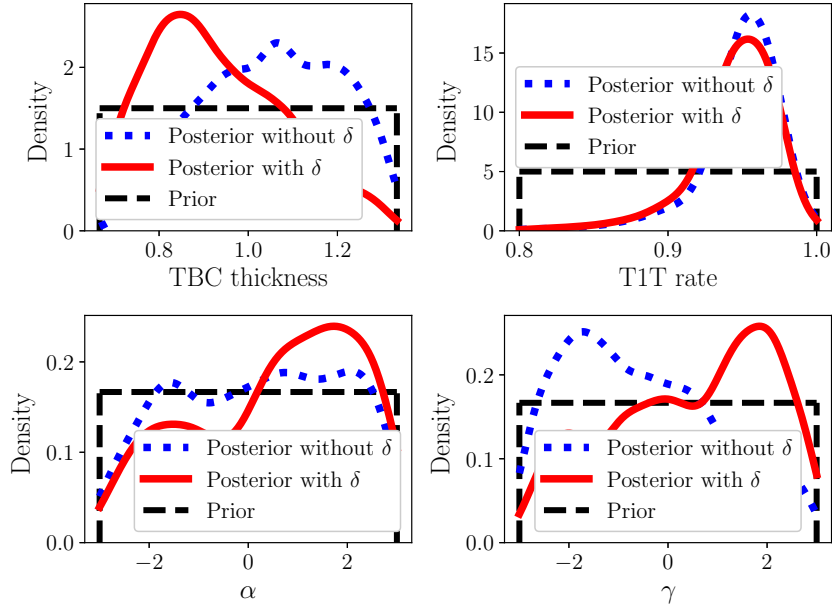


Figure 6.11: Posterior distributions of model parameters using elongation data at the blade tip (blade ID 90), using online Bayesian calibration

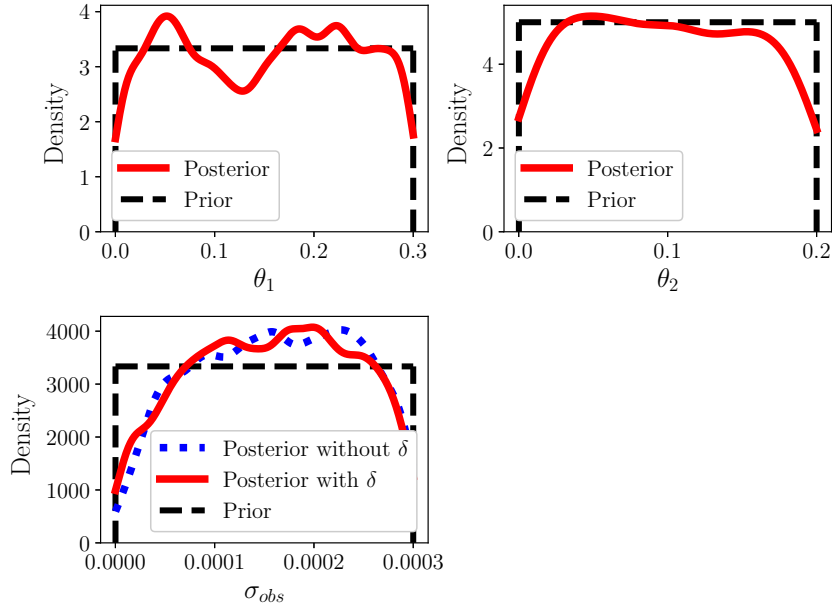


Figure 6.12: Posterior distributions of noise standard deviation and model discrepancy hyperparameters using elongation data at the blade tip (blade ID 90), using online Bayesian calibration

and 3D deformation). On the other hand, in many applications online calibration is crucial in identifying any changes in real-time. Moreover, the online calibration shows significant computational improvements in parameter estimation and prediction. The computational times of the entire offline

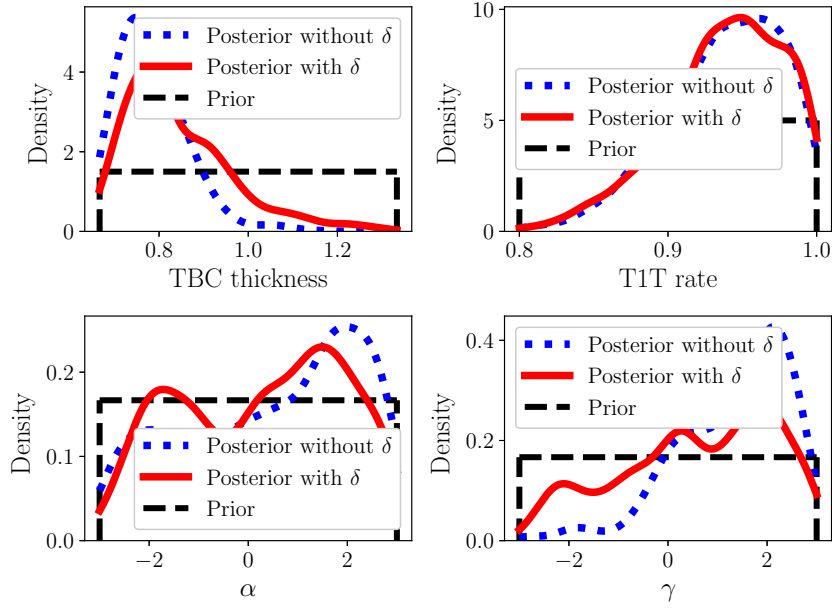


Figure 6.13: Posterior distributions of model parameters using 3D deformation data of blade ID 01, using online Bayesian calibration

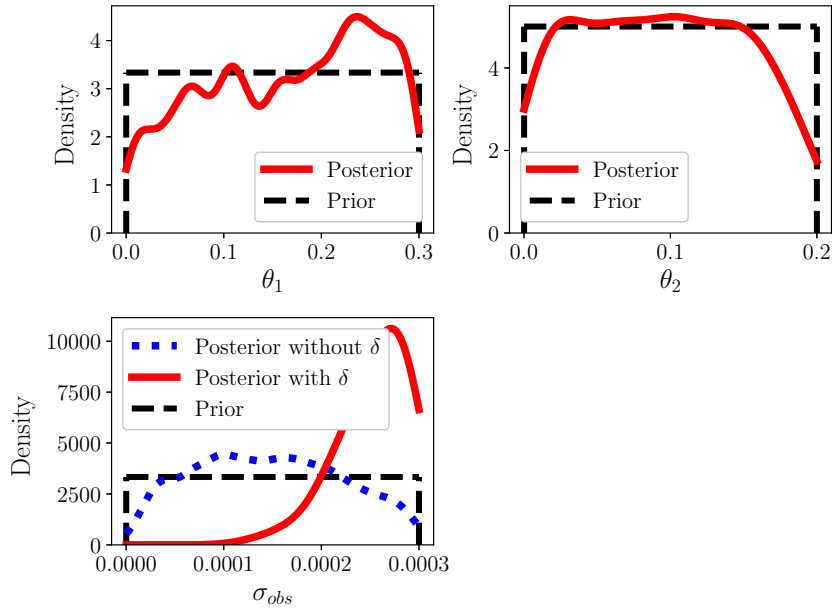


Figure 6.14: Posterior distributions of noise standard deviation and model discrepancy hyperparameters based on 3D deformation data of blade ID 01, using online Bayesian calibration

and online calibration (i.e., multi-level calibration using different types of measurement data) with 10,000 particles are approximately 48 and 27 minutes on a desktop with an Intel 8-core CPU with a 3.00 GHz base frequency and 8 GB memory, which includes surrogate model predictions for each

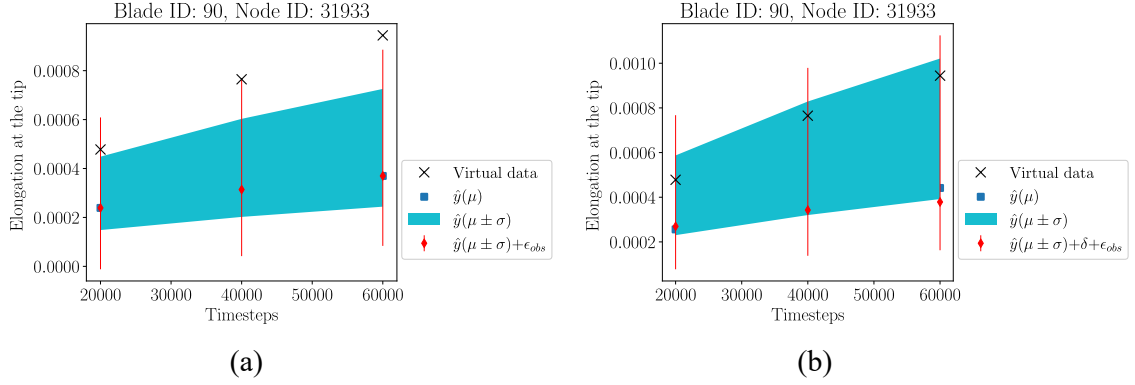


Figure 6.15: Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the elongation at the blade tip data of blade ID 90, using online Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy

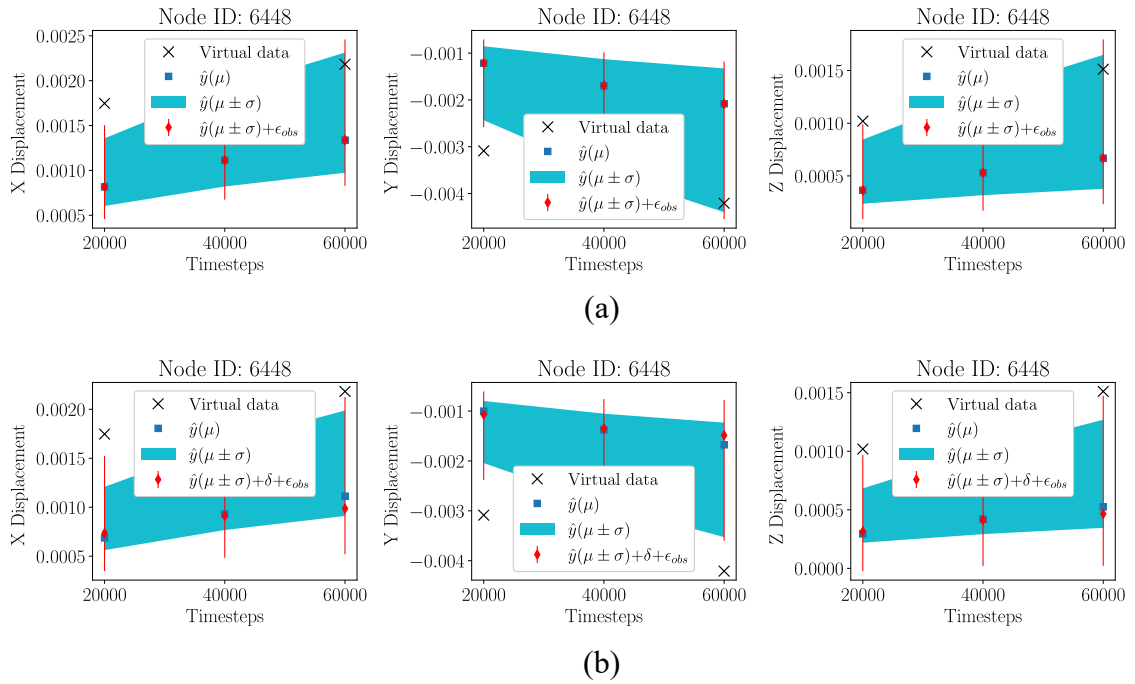


Figure 6.16: Prediction \hat{y} at the mean and plus or minus one standard deviation of the calibrated parameters using the 3D deformation data of blade ID 01, using online Bayesian calibration: (a) Without model discrepancy, (b) With model discrepancy

particle at each time step where measurements are taken.

6.4 Summary

This chapter presented a Bayesian calibration framework for time-dependent multi-component systems. The proposed framework fuses multi-physics models with sensor data, and considers different uncertainty sources. The main components of the proposed framework are the development

of the offline and online Bayesian calibration strategies for multi-source information fusion, calibration of uncertain parameters, uncertainty reduction and handling of variables for the system level prediction.

A surrogate model is first built to replace the expensive physics-based FEM for faster calibration; this surrogate model is built to produce a very high-dimensional output of multiple QoIs over a large number of spatial locations and time instants. Once the surrogate model is considered satisfactory, we perform Bayesian calibration to infer model parameters and the model discrepancy term based on measurement data. A multi-level Bayesian calibration approach is investigated using a Hierarchical Bayesian Network (HBN) by fusing different data types from different components at different time steps. This approach helps to incorporate all the available information into the calibration strategy. The uncertainty regarding local parameters is reduced by using different types of information at multiple levels. Hence, the multi-source information fusion enhances the uncertainty reduction in the unknown parameters.

The important features of the calibration approach can be summarized as follows:

- The unknown parameters are calibrated with multi-component data using the iterative strategy described in Section 6.2.2.
- When there exist local parameters (i.e., component-level) and global/shared parameters (i.e., system-level), these parameters are calibrated multiple times as the posteriors from the lower-level calibration is used as the priors for the higher-level calibration.
- The local parameters corresponding to the component where data is available are re-calibrated using the single-component data.

Both offline and online calibration strategies were investigated. The offline strategy, where the multi-component calibration (i.e., the lower-level BN) performed with data from all time steps (unlike the online strategy), converges to sharper posterior distributions. Since the posterior estimates obtained at the lower-level BNs are used as priors at the higher-level BNs, the posterior distributions of local parameters at the higher-level BNs are sharper.

CHAPTER 7

Decision-Making under Uncertainty

7.1 Introduction

A large system is often designed to meet multiple objectives, which in many cases are conflicting with each other. In order to improve the system performance, the conflicting objectives need to be balanced through multi-objective optimization. Also, since system variability is a major concern and there is model uncertainty in the predicted quantity of interest (QoI), it is necessary to optimize the process parameters for multiple objectives while also considering these uncertainty sources. This objective develops a framework for optimizing process parameters under uncertainty and investigates whether the limitations of physics-based models and single-objective optimization could be overcome using ML models and a multi-objective optimization approach.

The introduction of system models (either physics-based or data-driven) introduces several sources of uncertainty such as model parameter uncertainty, input uncertainty, model error, etc., which is then propagated to the QoI predicted by the model [26, 32, 38]. Data-driven models are created with data collected from experiments. The uncertainty in the measurement process from instruments, human error, etc. needs to be considered. It is also important to select the optimal model and tuning parameters of the model (e.g., number of layers and units in a deep neural network), and avoid data overfitting. Often the amount of data available to construct the data-driven model is limited, leading to uncertainty in the model prediction. Therefore, it is necessary to consider the model uncertainty for an accurate and reliable prediction model. The presence of input uncertainty and model uncertainty introduces uncertainty in the prediction of the model outputs. Such stochastic optimization formulation often suffers from intensive computational effort since optimization under uncertainty requires an extra loop of uncertainty quantification (UQ) in each optimization iteration. Thus, surrogate modeling techniques, which replace the expensive physics code with an inexpensive model for UQ, are often used for optimization under uncertainty.

A generic formulation of deterministic multi-objective optimization for n_{obj} objectives may be written as:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} && \{f_1(\mathbf{x}, \mathbf{p}), \dots, f_{n_{obj}}(\mathbf{x}, \mathbf{p})\} \\ & \text{subject to} && g_i(\mathbf{x}, \mathbf{p}) \leq 0, \quad i = 1, 2, \dots, n_{con} \\ & && \mathbf{lb}_x \leq \mathbf{x} \leq \mathbf{ub}_x \end{aligned} \tag{7.1}$$

where \mathbf{x} and \mathbf{p} are the design and non-design variables, g_i , $i = 1, 2, \dots, n_{con}$ represent the n_{con} deterministic constraints, and $\mathbf{lb}_\mathbf{x}$ and $\mathbf{ub}_\mathbf{x}$ are the lower and upper bounds for the design variables \mathbf{x} . The solutions of multi-objective optimization are usually characterized by a Pareto front, which is a series of designs describing the trade-off among different objectives. To construct the Pareto front, four approaches have been studied in the literature: weighted sum, constraint-based methods, goal programming, and genetic algorithm. The decision maker can choose the appropriate design based on the preferences on the objectives [162].

Optimization under uncertainty can be pursued in two directions: (1) robust design optimization (RDO) [163], and (2) reliability-based design optimization (RBDO) [164]. In RDO, both the mean and the variability of the objective function are optimized (since minimizing the variability makes the objective insensitive to variations of the input variables and parameters), and the constraints are satisfied within specified uncertainty bounds. On the other hand, in RBDO, a desired target level of reliability is maximized (i.e., probability of satisfying a desired threshold of performance or quality) by optimizing the decision variables, or a cost function is minimized while satisfying a reliability constraint.

Consider the case where the objective is to minimize a function $f(\mathbf{x}, \mathbf{p})$ with design variables $\mathbf{x} = [\mathbf{x}_d, \mathbf{x}_\theta]$ and non-design variables $\mathbf{p} = [\mathbf{p}_d, \mathbf{p}_\theta]$, where $\mathbf{x}_d, \mathbf{p}_d$ are deterministic variables, and $\mathbf{x}_\theta, \mathbf{p}_\theta$ are stochastic design and non-design variables respectively. The single objective RDO formulation using weighted sum approach can be written as

$$\begin{aligned}
& \underset{\mathbf{x} \in \mathbb{R}^{n_x}}{\text{minimize}} && w_1 \mu_f(\mathbf{x}, \mathbf{p}) + w_2 \sigma_f(\mathbf{x}, \mathbf{p}); \\
& \text{subject to} && \mathbf{lb}_\mathbf{g} + \mathbf{k}_c \sigma_\mathbf{g} \mathbf{g}(\mathbf{x}, \mathbf{p}) \leq \mathbf{g}(\mathbf{x}, \mathbf{p}) \leq \mathbf{ub}_\mathbf{g} - \mathbf{k}_c \sigma_\mathbf{g} \mathbf{g}(\mathbf{x}, \mathbf{p}), \\
& && \mathbf{lb}_\mathbf{x} \leq \mathbf{x} \leq \mathbf{ub}_\mathbf{x}
\end{aligned} \tag{7.2}$$

where $\mu_f(\cdot)$ and $\sigma_f(\cdot)$ are the mean and standard deviation of f , $0 \leq w_1, w_2 \leq 1$ are the weights representing the relative importance of each objective function, \mathbb{R}^{n_x} represents the design space, $\mathbf{lb}_\mathbf{x}, \mathbf{ub}_\mathbf{x}$ represent the lower and upper bounds for the design variables, $\mathbf{g}(\mathbf{x}, \mathbf{p})$ is the vector of inequality constraints, and $\mathbf{lb}_\mathbf{g}, \mathbf{ub}_\mathbf{g}$ represent the lower and upper bounds of the constraints. For stochastic constraints, the feasible region is reduced by $k_c \sigma_g$ in each direction where \mathbf{k}_c is a vector of user-defined constants based on the design requirements and σ_g is the vector of standard deviations of the constraints [163]. For a deterministic constraint, $k_c \sigma_g = 0$. Note that in this work, the optimization needs to consider multiple objectives, considering multiple QoIs. The same approach in the above equation can be extended for n_{obj} objectives by assigning corresponding weights

$w_1, w_2, \dots, w_{n_{obj}}$ to the objective functions. Depending on the design variables and constraints, the optimization problem discussed above can be solved using a suitable global optimization algorithm. A brute-force optimization may be adequate for optimization problems with small number of design variables with small ranges. However, when the design space cannot be as easily explored, other optimization techniques such as genetic algorithm [165], simulated annealing [166], particle swarm [167], Non-dominated Sorting Genetic Algorithm II (NSGA-II) [168], etc. can be employed.

7.2 Single Physical Quantity of Interest^{1,2}

Many previous studies on process optimization in AM have employed a design of experiments-based trial-and-error approach to determine the effect of different process parameter settings on the QoI such as part strength, residual stress, surface roughness etc. [89, 169–171]. The experiment-based approach is problem-specific, and cannot be used as a general-purpose process design method for different materials and part geometries. Therefore, there is a need to use a process model (either physics-based or data-driven) that optimizes the process parameters for better overall part quality without running multiple economically expensive physical experiments, and also includes the uncertainty in the model prediction. The models used to predict the QoI are affected by various sources of uncertainty and error. This affects process optimization and process control decisions. Thus, it is critical to identify and quantify the effects of the uncertainty and error sources in order to improve the overall quality of FFF products. This chapter develops a framework for optimizing process parameters under uncertainty that maximizes the filament bond quality of FFF printed parts.

The quality of an additively manufactured part is related to the process parameters such as printer nozzle temperature, nozzle speed, layer width, layer height etc. The process parameters can be more easily controlled than other factors such as the material properties of the raw material. The effect of process parameters on the QoI has been studied extensively. Armillotta et al. [172] investigated the effect of included angle, inclination, and incidence angle on the edge quality of FFF parts. Chacón et al. [173] analyzed the effect of different process parameters on the mechanical properties of the printed parts. The relationship between dimensional accuracy and infill percentage, infill pattern, layer thickness, and extrusion temperature is studied by Alafaghani and Qattawi [174] using Taguchi design of experiment method. These studies use a problem-specific experiment-based trial-and-error approach to select the process parameters. This approach is inefficient and costly. In addition, the

¹Adapted with permission from: [Kapusuzoglu, B., Sato, M., Mahadevan, S., & Witherell, P., “Process Optimization Under Uncertainty for Improving the Bond Quality of Polymer Filaments in Fused Filament Fabrication,” ASME. J. Manuf. Sci. Eng., vol. 143\(2\), no. 021007, 2021.](#)

²© 2020 by ASME

experimental results are not generalizable to a different experimental setting, component, or design and the trial-and-approach cannot be used to make decisions for unexplored scenarios.

The focus of this section is to develop a framework for optimizing process parameters under uncertainty that maximizes the filament bond quality of FFF printed parts. The overall bond length is the decision criterion we use for optimization; greater overall bond lengths signify lesser void area (thus better bonding) in an average sense. We only consider stage 1 and 2 shown in Fig. 7.1 to predict the overall bond length. Stage 3 is not considered in the bond formation modeling. Other alternative criteria could also be used within this framework, such as reducing the void area. Note that the focus of this chapter is not to improve the process, but to make optimum use of the existing process. In this context, the proposed methodology leverages the transient heat transfer model and sintering neck growth model for cylindrical filaments proposed by Costa et al. [175] and Gurrala et al. [80], respectively. It quantifies the uncertainty in the coupled multi-physics model prediction, and determines the optimal process parameters through model-based optimization under uncertainty to improve the bond quality at each layer to overcome poor intra-layer and inter-layer bonding. Variance-based sensitivity analysis based on Sobol’ indices is used to quantify the relative contributions of different uncertainty sources to the uncertainty in the bond quality. The uncertainty in the coupled multi-physics model prediction is quantified by constructing a Gaussian process (GP) surrogate model to compute and include the model discrepancy within the optimization. Physical experiments are conducted for calibration and validation of the physics model, and also for validation of the optimum solution.

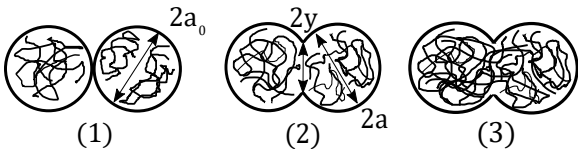


Figure 7.1: The bond formation process between two filaments: (1) initial surface contact; (2) wetting or neck growth; (3) molecular diffusion and randomization across the cross-section of two FFF extruded filaments

In summary, the contributions of this section are as follows:

1. Development of a Bayesian methodology to quantify the uncertainty in the neck growth prediction.
2. Construction of a GP surrogate model for efficient computation of model error, in order to incorporate model error in process parameter optimization.

3. Evaluation of the relative contributions of various uncertainty sources to the uncertainty in the model output.
4. Development of a computational framework for process parameter optimization under uncertainty in order to maximize the bond quality between extruded polymer filaments in FFF.

7.2.1 Proposed Methodology

The proposed methodology for process parameter optimization under uncertainty consists of the following steps:

1. Uncertainty quantification in FFF product bond quality
2. Probabilistic sensitivity analysis
3. Surrogate modeling of physics model discrepancy
4. Process parameter optimization under uncertainty
5. Physical experiments for model calibration and validation

The following subsections describe these steps in detail.

7.2.1.1 Uncertainty Quantification in FFF

In this subsection, several uncertainty sources in FFF are identified and methods to quantify them are discussed. The heat transfer and sintering neck growth models used in this section have their own model inputs, parameters, and errors. Some of these model inputs, parameters, and errors are deterministic while others are uncertain. The uncertainty sources can be aleatory (natural variability) or epistemic (lack of knowledge). The input values used in the heat transfer model (the thickness, width and length of filaments, printer nozzle temperature and extrusion speed) may not be the same as the actual value, thus introducing uncertainty regarding the input to the bond length model. Of the above parameters, the printer nozzle temperature is assumed to vary across printed parts. The temperature of the filaments immediately after being extruded was found to be significantly lower than the specified printer nozzle temperature. This variation in the temperature of the filament as it leaves the nozzle tip is included in the heat transfer model. Moreover, the heat transfer model is also affected by model parameter uncertainty, which is considered epistemic uncertainty (i.e., they have fixed values which are unknown), such as density, specific heat capacity, convective heat transfer coefficient, and fractions of filament perimeter that is in contact with another filament

or with the build plate. Thus, the uncertainty in the printer nozzle temperature and model parameters of the heat transfer model introduces uncertainty in the temperature history of the interfaces. The uncertainty in the output of the heat transfer model further propagates to the output quantity of interest through the sintering neck growth model. In addition, the uncertainty regarding the model parameters of the sintering model (such as surface tension, and material viscosity) introduce additional uncertainties in the quantity of interest. Both models also have errors since they have various assumptions, and are not perfect representations of the actual physics. Thus, the bond length predictions have uncertainty due to the propagation of the effects of these different uncertainty sources. The unknown model parameters and errors can be estimated using the experimental data, which contain measurement noise/observation error, through Bayesian model calibration.

7.2.1.1.1 Model Calibration under Uncertainty

As discussed in Section 2.3, the model predictions are affected by model errors due to missing physics or approximations. Therefore, a model discrepancy term $\delta(\mathbf{X})$ as a function of model inputs (one of the main features of the Bayesian calibration framework developed by Kennedy and O’Hagan [50]) can be introduced as shown in Fig. 2.2 to capture the disagreement between the true system response and the model prediction. Input variables \mathbf{X} are measurable quantities and chosen by the experimenter. These can be considered deterministic or stochastic with known probability distributions due to natural variability (aleatory) or measurement error. Whereas, model parameters are uncertain due to lack of knowledge (epistemic) since θ_m take some unknown deterministic values during the experiment. The purpose of Bayesian model calibration is to use observation data \mathbf{Y}_{obs} to estimate the posterior distributions of θ_m and other unknown quantities such as parameters of observation error and the discrepancy term as discussed in Section 2.3. Kennedy and O’Hagan [50] employ a probabilistic relationship between the predictions and observations, which incorporates both model parameters and a discrepancy function. Note that the discrepancy function is not observable from the observation data, since the true values of model parameters are unknown. The model discrepancy function is treated as a Gaussian process (GP). The hyperparameters of the GP (including the coefficients of the trend function) can be estimated along with physics model parameters using a Bayesian approach. However, in the presence of insufficient amount of experimental data and non-informative prior knowledge about the uncertainty sources in the engineering system, it may be difficult to distinguish between the effects of the model parameters and model discrepancy; this problem is referred to as non-identifiability [51, 161] when the number of parameters and hyperparameters that need to be estimated becomes large when the model discrepancy term is treated

as a GP.

Two strategies are pursued here for improving the identifiability: (1) performing sensitivity analysis to identify the most important physics model parameters as described in Section 2.4, and (2) ignoring the discrepancy term during the calibration step and building a surrogate model for the discrepancy (i.e., the difference between the calibrated model prediction and actual system response), as described in Section 2.2.1.

First, the physics model parameters that have the most significant contribution to the uncertainty in the model output during the entire printing process are calibrated without including the model discrepancy term. The joint posterior distribution of θ_m can be computed using Bayes' theorem (see Eq. (2.20)). Bayesian model calibration is often performed using Markov chain Monte Carlo (MCMC) sampling algorithms (such as Metropolis-Hastings [55], Gibbs [56], or slice sampling [57]) since the integral in the denominator of Eq. (2.20) makes numerical integration intractable for increasing dimension of calibration quantities [58]. The Metropolis-Hastings algorithm is used in this chapter.

Next, a GP surrogate model is constructed as discussed in Section 2.2.1 for the model discrepancy. A set of additional experiments can be conducted to obtain training data for building a surrogate model such as GP model in order to estimate the model discrepancy at any input value \mathbf{x} . (Note that two sets of experimental data are used: the first set for calibrating the physics model parameters and the second set for building the surrogate model of model discrepancy. This two-step approach was possible in this study since the experiments were inexpensive and fast; if it is not possible to conduct two sets of experiments, then simultaneous calibration of physics model parameters and GP hyperparameters will need to be pursued, with appropriate assumptions and sensitivity analysis to reduce the number of calibration quantities and achieve identifiability). The training data of the GP model for the model discrepancy can be evaluated for different input values of experimental tests and realizations of observation errors by comparing model predictions against experimental data $\mathbf{y}_D(\mathbf{x})$

$$\delta(\mathbf{x}) = \mathbf{y}_D(\mathbf{x}) + \epsilon_{\text{obs}} - \mathbf{y}_m(\mathbf{x}). \quad (7.3)$$

The GP model ($\delta_{GP}(\mathbf{x})$) for the model discrepancy captures the combined contribution of model form and measurement error for a given bond length. Thus, the corrected prediction of the physics-based model \mathbf{y}_{pred} can be written as

$$\mathbf{y}_{\text{pred}}(\mathbf{x}) = \mathbf{y}_m(\mathbf{x}) + \delta_{GP}(\mathbf{x}). \quad (7.4)$$

Two cases with different inputs and model functions are considered in this section, namely cases A and B. The inputs to the sintering neck growth model, temperature profiles of the extruded filaments, are predicted using the heat transfer model in case A, whereas, measured temperature data from experiments are used as the inputs to the neck growth model in case B. $\mathbf{G}(\mathbf{x}; \theta_m(\mathbf{x}))$ represents (a) the coupled physics-based heat transfer and sintering neck growth models for case A and (b) sintering neck growth model, which can be evaluated using a numerical technique such as 4th order Runge-Kutta method, for case B (see Fig. 7.2). In these cases, the physics model is inexpensive to evaluate, thus a surrogate model has not been built for the model $\mathbf{G}(\mathbf{x}; \theta_m(\mathbf{x}))$.

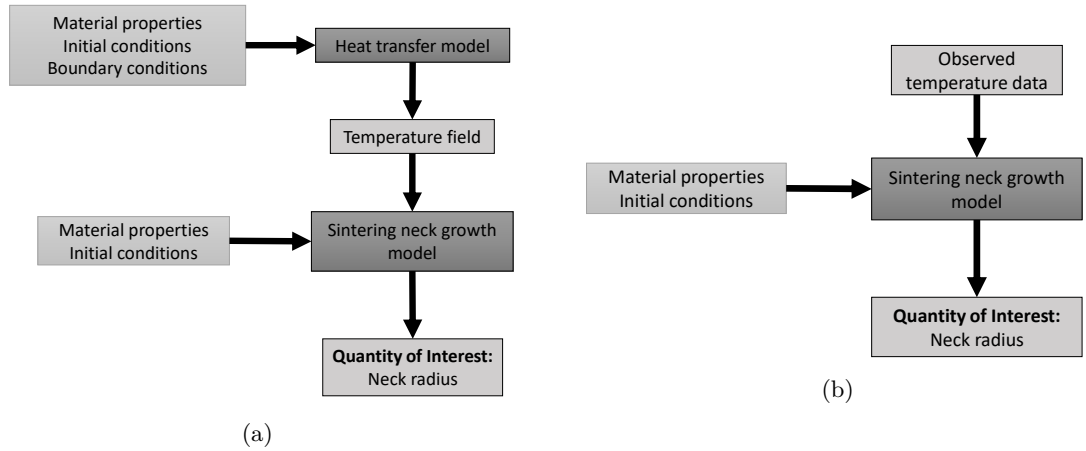


Figure 7.2: Flowchart of the simulation models for (a) Case A and (b) Case B

7.2.1.2 Process Optimization under Uncertainty

The optimal design point that satisfies design criteria and a specified level of reliability is of great interest in many engineering applications. In this section, the focus is on selecting the optimum values of process parameters that maximize the bond lengths between filaments and between layers. The AM process optimization under uncertainty can be pursued in two directions: (1) reliability-based design optimization (RBDO) [164], and (2) robust design optimization (RDO) [163]. In RBDO, the decision variables are optimized to either maximize or achieve a desired target level of reliability (i.e., probability of satisfying a desired threshold of performance or quality). In RDO, the decision variables are optimized such that the variability of the objective function is minimized, and the constraints are satisfied within specified uncertainty bounds. The approach of robustness-based design optimization (RDO) is used to maximize the overall bond quality. The robustness of the objective function can be achieved by simultaneously optimizing the mean and variance; thus, this is a bi-objective problem. Monte Carlo sampling is used to compute the mean and variance of

the objective function (the mean bond length of a layer) in the probabilistic optimization process. An efficient sampling-based method Latin hypercube sampling is used to simulate the uncertain parameters.

The robust design optimization problem can be formulated as follows:

$$\begin{aligned}
& \underset{\mathbf{d} \in \mathbb{R}^{n_d}}{\text{minimize}} && \{E(y_o(\mathbf{d})), V(y_o(\mathbf{d}))\}; \\
& \text{subject to} && \{E(y_c(\mathbf{d})), V(y_c(\mathbf{d}))\} \leq 0, \quad c = 1, 2, \dots, n_c; \\
& && d_{i,\text{lb}} \leq d_i \leq d_{i,\text{ub}}, \quad i = 1, 2, \dots, n_v,
\end{aligned} \tag{7.5}$$

where the objective function and constraints are expressed as functions of expectation $E(\cdot)$ and variance $V(\cdot)$ of objective function y_o and constraints y_c , respectively; the vector of design variables \mathbf{d} can be either deterministic parameters or mean/standard deviation of the design parameters, n_c and n_v are the number of constraints and design variables, respectively. The i -th design variable d_i is constrained by its lower bound $d_{i,\text{lb}}$ and upper bound $d_{i,\text{ub}}$. The RDO problem becomes bi-objective by adding the variance of the performance function to the expected value of the objective function, and a weighted sum method can be used to assign proportional weights for the aggregation of the two objectives according to their importance [176]. The aggregation formulation is given by Eq. (7.2).

The nested bi-objective robustness-based design optimization (RDO) problem can be converted into a single objective formulation, using a weighted sum approach, as:

$$\begin{aligned}
& \underset{\mathbf{x} \in \mathbb{R}^{n_x}}{\text{minimize}} && w_1 F_1 + w_2 F_2 \\
& \text{subject to} && \mathbf{x}_{\text{lb}} \leq \mathbf{x} \leq \mathbf{x}_{\text{ub}}
\end{aligned} \tag{7.6}$$

where weighting coefficients $w_1, w_2 > 0$ represent the relative importance of two objectives. $F_1 = -\mu_{\mu_{\text{BL},i}} + \sigma_{\mu_{\text{BL},i}}$ represents the mean and standard deviation of the average bond length predictions μ_{BL} for layer i , and $F_2 = \mu_{\sigma_{\text{BL},i}} + \sigma_{\sigma_{\text{BL},i}}$ represents the mean and standard deviation of the standard deviation of the bond length predictions σ_{BL} for layer i , $i \in \{1, \dots, M\}$. M is the total number of layers, \mathbf{x} are the design variables (printer nozzle temperature and printer extrusion speed for each layer i), and $\mathbf{x}_{\text{lb}} \leq \mathbf{x} \leq \mathbf{x}_{\text{ub}}$ represents the lower and upper bounds for the design variables. The weighted sum approach is a convex combination of two different objectives, F_1 and F_2 . The solution of the optimization problem approximates the Pareto front by changing the weights of each objective. The Pareto front maps the relation between these two objective functions. The negative of the mean value of mean bond lengths at each layer is minimized while minimizing the deviation of mean bond

lengths at each layer with the use of function F_1 . The function F_2 , which is a convex combination of the mean and standard deviation of the deviation of bond lengths at each layer, is minimized simultaneously with F_1 . In other words, the overall bond quality (the mean value of bond length for a part) is maximized, while minimizing the variations in the quantity of interest (bond lengths between filaments) using the functions F_1 and F_2 respectively.

7.2.1.3 Experimental Work

A commercial material, Ultimaker Black ABS, was used in the experiments. A unidirectional and aligned building strategy was adopted at a specified printer nozzle temperature and extrusion speed. Two different options for the deposition sequence of the filaments were considered, as shown in Fig. 7.3. In Fig. 7.3a, the filaments are sequenced from left to right in all the layers. In Fig. 7.3b, the filaments are sequenced from left to right in odd numbered layers and from right to left in even numbered layers. The filament numbers in the two figures correspond to the two deposition sequence options.

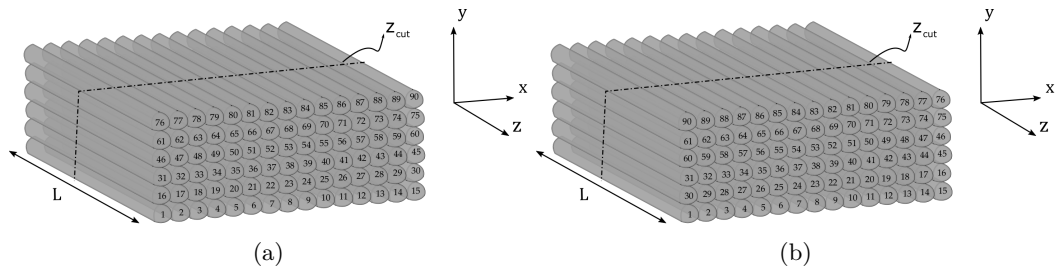


Figure 7.3: Deposition sequence of unidirectional 90 filaments: (a) from left to right for all the layers and (b) from left to right in odd numbered layers and from right to left in even numbered layers

Multiple rectangular-shaped specimens were produced with the same geometry but different combinations of process parameter values. For each specimen, the temperature distribution at the top of each layer during deposition was monitored using an infrared thermography camera. Thermal images were recorded with a specified frequency until all filaments were deposited. The neck growth between the filaments and the total void area of the parts were identified at a specified cross-section with the use of microscopy images processed through the ImageJ software [88]. The statistical properties of the neck growth along the length of the specimens were constant. Therefore, all specimens were sectioned at the midpoint to analyze the mesostructural feature of interest only at that cross-section.

7.2.1.4 Numerical Example

The experimental setup used to build rectangular acrylonitrile butadiene styrene (ABS) amorphous polymer specimens of length 35 mm, width 12 mm, and thickness 4.2 mm is shown in Fig. 7.4. The specimens were created on an Ultimaker 2 extended+ printer, which is within an enclosure to reduce the part variability; a commercial material, Ultimaker Black ABS, was used. Air flow was not considered because we enclosed the printer to prevent the occurrence of airflow. All parts were printed through a nozzle with 0.8 mm diameter. The build plate temperature was constant and set to 110°C and the environment temperature is assumed to be 70°C. The extrusion rate and vertical position of the nozzle were adjusted by the printer to be able to produce each filament with 0.8 mm width and 0.7 mm height.

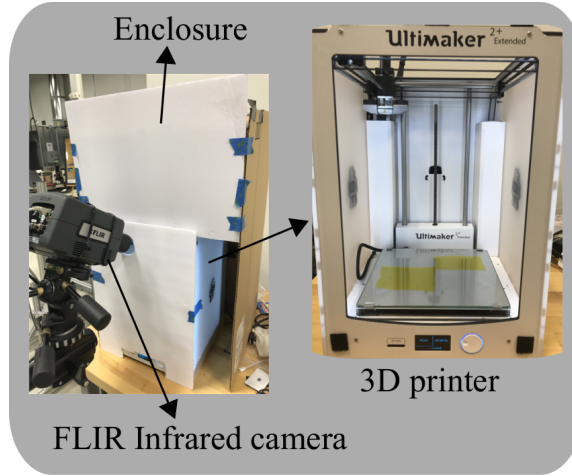


Figure 7.4: The experimental setup

The surface temperature profiles of extruded filaments were monitored using an infrared thermography camera as shown in Fig. 7.4. The extrusion of the next layer prevents the camera from monitoring the temperature profiles of the previous layers. Due to the inability to obtain temperature data of the filaments below the top layer, we could only predict the quality of the intra-layer bonding using temperature profiles of extruded filaments within that layer. Thermal images were recorded with a frequency of 10 Hz until the deposition of all filaments was completed.

All specimens used in this study are produced with unidirectional filaments to enhance the effects of process parameters on the bond quality between adjoining filaments. Each filament of the rectangular part is deposited at a specified printer nozzle temperature T_n and extrusion speed v_p . The temperature evolution of the interfaces, and the neck growth between the filaments (the mesostructural feature of interest), are predicted at $z_{\text{cut}} = L/2$ as shown in Fig. 7.3. The process parameters and material properties used in this work are presented in Table 7.1. The specific heat

capacity and density of the material are calibrated together as a single term $\alpha = \rho C$, where ρ and C are density (kg/m^3) and specific heat capacity ($\text{J/kg } ^\circ\text{C}$) respectively. The analysis assumes temperature dependent material properties such as material viscosity η and surface tension Γ . The surface tension of ABS P400 at 240°C is 0.029 N/m as reported by Bellehumeur et al. [78] with a temperature dependence $\Delta\Gamma/\Delta T = -\gamma \text{ N/m} \cdot \text{K}$, where the neck growth model parameter $\gamma = 0.00345$. The temperature dependent material viscosity η is given by $\eta = \eta_r \exp[-\beta(T - T_r)]$, where the material viscosity at the reference temperature ($T_r = 240^\circ\text{C}$) η_r is $5100 \text{ Pa} \cdot \text{s}$, β is a model parameter that is selected as 0.056 by Sun et al. [89], and T is the temperature of the material at a given time instance.

Table 7.1: Process parameters and material properties

Property	Value
Printer nozzle temperature ($^\circ\text{C}$)	240
Build plate temperature ($^\circ\text{C}$)	110
Printer extrusion speed (m/s)	0.042
Filament length (m)	0.035
Filament width (m)	0.0008
Filament thickness (m)	0.0007
Fraction of filament's perimeter for all contacts	0.15
Convective heat transfer coefficient ($\text{W/m}^2 \text{ } ^\circ\text{C}$)	86
Conductive heat transfer coefficient between filaments ($\text{W/m}^2 \text{ } ^\circ\text{C}$)	200
Conductive heat transfer coefficient between filament and build plate ($\text{W/m}^2 \text{ } ^\circ\text{C}$)	86
Thermal conductivity ($\text{W/m } ^\circ\text{C}$)	0.15
α ($\text{J/m}^3 \text{ } ^\circ\text{C}$)	1.196×10^6

7.2.1.4.1 Prediction of the Cooling of Filaments

A typical IR image of the temperature profile for the first layer of a part printed using $(T_n, v_p) = (240^\circ\text{C}, 0.042 \text{ m/s})$ is shown in Fig. 7.5. The interface temperature is monitored at corresponding locations between filaments. The experimental temperature profile was used to assess the validity of the heat transfer model in order to be able to predict the neck growth accurately using the heat transfer model predictions. The temperature of the filaments immediately after being extruded onto the build plate or onto another filament was found to be significantly lower (20°C to 50°C) than the specified printer nozzle temperature. At the upper temperature limit of the printer the filaments were extruded at temperatures approximately $40\text{-}50^\circ\text{C}$ less than the set nozzle temperature. The enclosure was not a precisely controlled environmental chamber with quantitative measurements of temperature, humidity, and airflow. Therefore, these effects might be influencing the difference between the printer setting and observed temperature. This variation in the temperature of the filament as it leaves the nozzle tip is considered as a bias term in the heat transfer model.

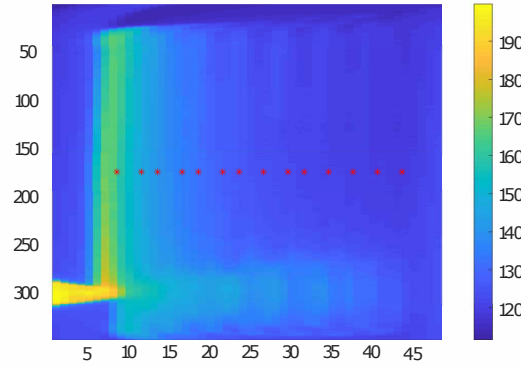


Figure 7.5: Top view temperature profile of the first layer

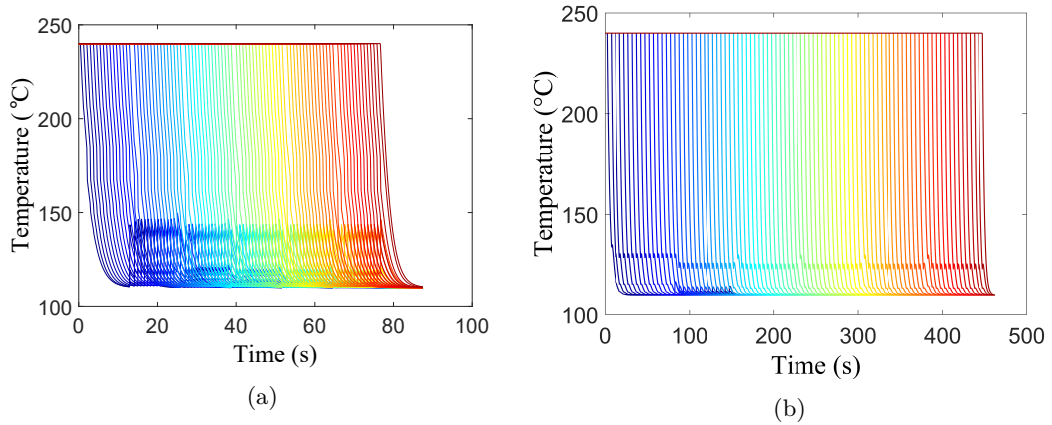


Figure 7.6: Temperature evolution of 90 filaments at $z_{\text{cut}} = L/2$ (a) with $L = 0.035$ m shown in Fig. 7.3a and (b) with $L = 0.21$ m shown in Fig. 7.3b along deposition time

The temperature evolution of all the filaments illustrated in Fig. 7.3a and Fig. 7.3b at $z_{\text{cut}} = 0.0175$ m along deposition time is shown in Fig. 7.6a and Fig. 7.6b respectively. The length to diameter ratio of filaments has a significant effect on the cooling process. The time it takes for the printer to extrude a single filament increases as the lengths of the filaments get longer. This results in a faster cooling process, and consequently a smaller amount of heat transfer between each filament. Moreover, extruding each layer's first filament at the same x -coordinate results in a more homogeneous part quality as the temperature difference between the filaments extruded on top of the filaments below is approximately the same. Whereas, in Fig. 7.3b, the temperature difference between the 1st and 30th filaments is much greater than 15th and 16th filaments, resulting in a staggered temperature evolution and part quality. Therefore, the build strategy shown in Fig. 7.3a is used for further analysis.

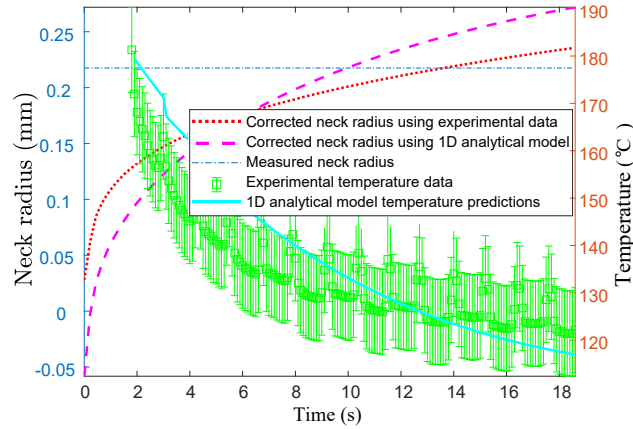


Figure 7.7: Experimental temperature profiles compared with model predictions for the interface between filaments 1 and 2 at $z_{\text{cut}} = 0.0175$ m and the neck growth predictions using the heat transfer model predictions (case A) and experimental temperature data (case B)

The Newtonian sintering model is coupled with the heat transfer model to predict the bond lengths between filaments. The one-dimensional transient heat transfer model predictions, and observed temperature data for the interface between filaments 1 and 2 are compared in Fig. 7.7. The model predictions are in general agreement with the measured data. The measured temperature data and model predictions show a similar trend in the initial stage due to enhanced convection at higher temperatures, but at temperatures below 130°C the model prediction deviates away from the measurement data. However, the inaccuracy of the model at lower temperatures is not relevant for neck growth predictions since the neck growth process occurs at higher temperatures [78].

The neck radius predictions for each case are corrected with the model discrepancy (estimated by the surrogate model) at given input values. The measured neck radius and the corrected neck growth predictions corresponding to case A (heat transfer model predictions as the input to the sintering neck growth model) and case B (observed temperature profile as the input to the sintering neck growth model) are demonstrated in Fig. 7.7 for $(T_n, v_p) = (240^{\circ}\text{C}, 0.042 \text{ m/s})$. The corrected neck radius predictions are in agreement with the measured data around 130°C since the model parameters are calibrated using the neck radius predictions when the interface temperature is at 130°C . Thus, case A is used for further analysis when the temperature data is not available.

7.2.1.4.2 Contribution Assessment of Uncertainty Sources

A sample-based single loop algorithm called MGSA (modularized GSA) proposed by Li and Mahadevan [101] is used to compute the first-order Sobol' indices. The results from MGSA indicate which parameters' individual effect have significant contribution to the uncertainty in the coupled

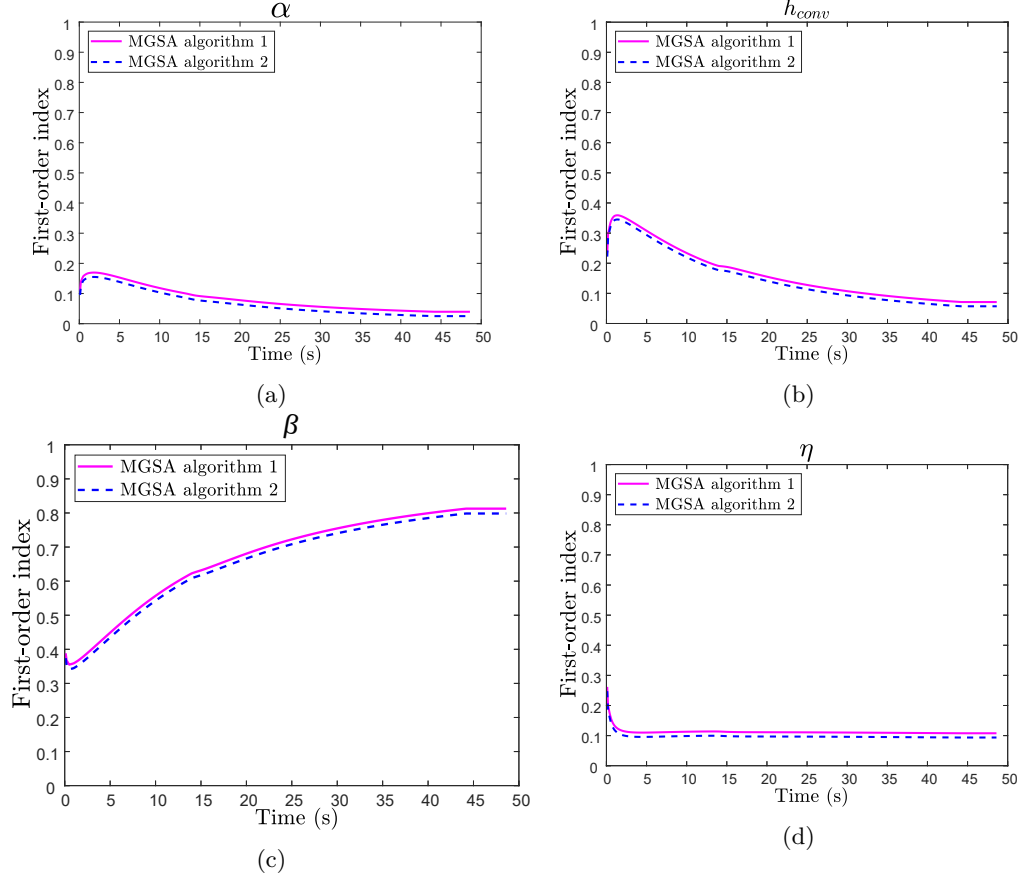


Figure 7.8: Sensitivity indices for the bond length between the 1st and 2nd filaments in the first layer

heat transfer and neck growth models. A low first-order index implies that the individual effect of the parameter is insignificant; thus, it can be fixed at its mean value. Thus, GSA provides insights on where to focus resources for improving the AM process.

The random variables in the heat transfer model are α, h_{conv} , and $\lambda_i, i \in \{1,2,3,4,5\}$, i.e., the material parameter, convective heat transfer coefficient and fraction of filament's perimeter that is in contact with other filaments or with the build plate. The random variables in the sintering neck growth model are Γ, η, β , and γ , i.e., the surface tension, material viscosity values at reference temperature of 240°C, and model parameters of the temperature dependent surface tension and material viscosity respectively.

7.2.1.4.2.1 GSA of the Bond Formation Model

The coupled heat transfer and sintering neck growth model considers eleven parameters as uncertain, i.e., $\alpha, h_{conv}, \lambda_i, i \in \{1,2,3,4,5\}, \Gamma, \eta, \beta$ and γ . As discussed earlier, the contributions of

various uncertainty sources to the neck growth vary for each layer as well; whereas, these contributions to the neck growth between filaments within a layer remain the same. Therefore, the first-order Sobol' indices of material parameters are assessed for four different neck growths at four different layers, i.e., the 1st, 20th, 50th and 87th bond formations in the first, second, fourth and sixth layers, respectively. We performed sensitivity analysis for each layer, at several interfaces within each layer. We present the results for only 4 layers (i.e., first, second, fourth and sixth layers) out of a total of 6 layers since the results corresponding to the fifth layer were the same as for the fourth and sixth layers, and the results corresponding to the third layer were the same as for the second layer. The results from GSA for these neck growths are illustrated in Figs. 7.8, 7.9, 7.10, and 7.11.

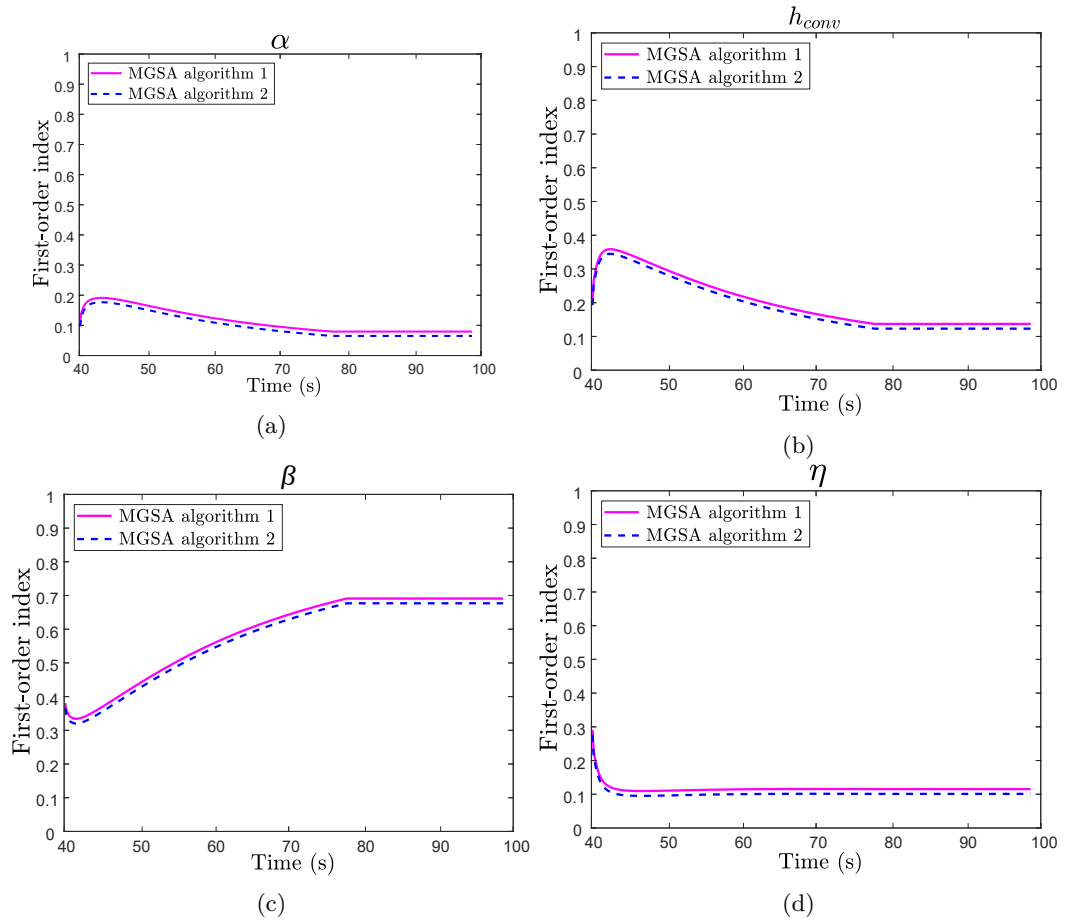


Figure 7.9: Sensitivity indices for the bond length between the 20th and 21st filaments in the second layer

The contributions of β increase significantly during the deposition of the specimen, while the contributions of α , λ_1 , λ_2 , λ_3 , λ_4 , λ_5 , h_{conv} , Γ , η and γ to the variations in the neck growth at given layers are negligible. The sensitivity index of β increases as the temperature of the interface between

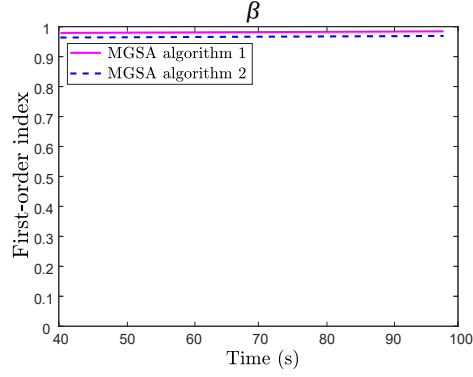


Figure 7.10: Sensitivity indices for the bond length between the 50th and 51st filaments in the fourth layer

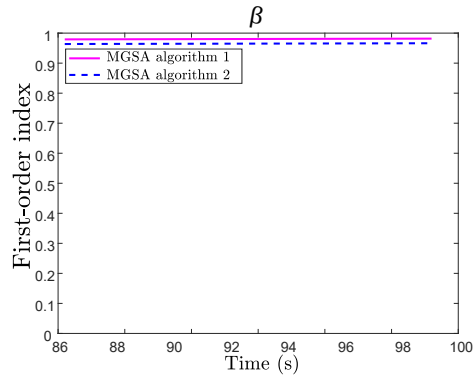


Figure 7.11: Sensitivity indices for the bond length between the 87th and 88th filaments in the sixth layer

the 1st and 2nd, and the 20th and 21st filaments cools down as illustrated in Figs. 7.8c and 7.9c.

The uncertainty grows fast as the deposition time increases or temperature decreases because the influence of uncertainty in the model parameter β on the neck growth increases with decreasing temperature. Another reason for the increase in uncertainty as the temperature decreases is that the neck growth model cannot capture the physics accurately when the temperature is below 130°C as shown in Fig. 7.7.

Based on the contributions of the various uncertainty sources to the neck growth at each layer of the part, β was found to be dominant for all the layers and interfaces considered; next was α (considering the high contribution of α to the uncertainty in the temperature evolution of filaments) which was found to be significant in some of the layers. All the other parameters are fixed at their mean value to reduce computational effort since their contribution to the uncertainty in the neck growth predictions were negligible compared to the parameters α and β , based on the rigorous sensitivity analysis described in Section 7.2.1.4.2.

7.2.1.4.3 Model Calibration

In a Bayesian setting, the epistemic uncertainty regarding the model parameters that have significant sensitivity indices can be reduced using experimental data. As discussed in Section 7.2.1.4.2, all material properties and model parameters, except α and β , are fixed at their nominal values. The measurement error is considered to be negligible since the measured bond lengths are precise to seven decimal points. Note that the material property α , which is required for the heat transfer analysis, is not needed in case B, where experimentally measured temperature profile is the input to the sintering neck growth model instead of the heat transfer model prediction.

The posterior distribution for the sintering neck growth model parameter β for case B (using observed temperature data as the input) is illustrated in Fig. 7.13. For case A, where heat transfer

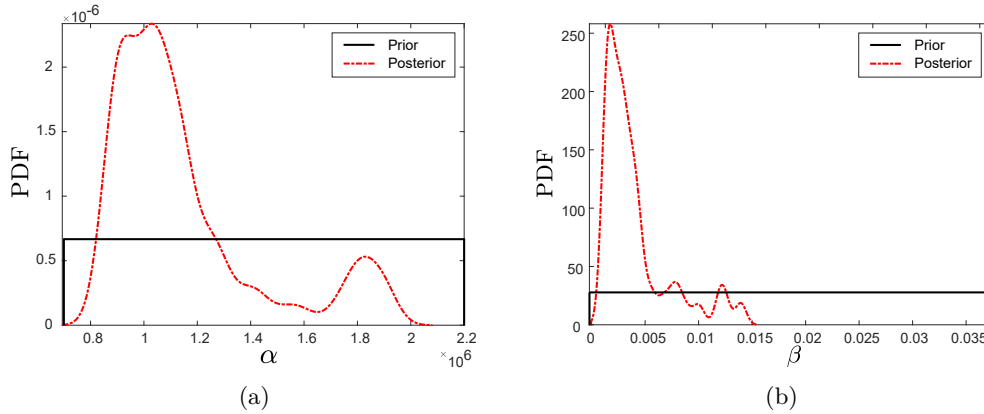


Figure 7.12: Prior and posterior distributions of the material property α and model parameter β considering the heat transfer model predictions as the input to the sintering neck growth model (case A)

model predictions are the input to the sintering neck growth model, the posterior distributions for α and β are shown in Fig. 7.12.

In order to calibrate these model parameters, 15,000 posterior samples are drawn using MCMC and the initial 5,000 samples are rejected (initial burn-in samples). The last 10,000 samples yield a mean value of 1.196×10^6 , and a coefficient of variation of 0.215 for $\alpha = \rho C$ and two different posterior distributions of β with mean values of 0.00378 and 0.0193, and coefficient of variations of 0.1154 and 0.05 for case A and B respectively. The mean of the posterior distribution of β for case B is close to the value reported in the literature [89]. However, for case A the uncertainty in α has a significant effect on the posterior distribution of β by shifting its mean to a smaller value. These values are used in the subsequent analysis (i.e., in the surrogate models for the model errors and optimization under uncertainty).

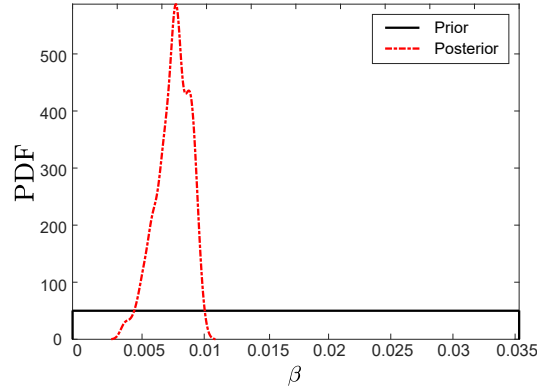


Figure 7.13: Prior and posterior distributions of the model parameter β using experimental temperature profile as the input to the sintering neck growth model (case B)

7.2.1.4.4 Surrogate Modeling

The surrogate models for the model discrepancy in cases A and B are built using the calibrated model parameters illustrated in Section 7.2.1.4.3. The GP model described in Section 2.2.1 is built with the training point inputs $[T_n, v_p, x, y]$ (i.e., nozzle temperature, printer speed, and the location of midpoint of the intra-layer bonds, respectively), and the corresponding intra-layer bond length model discrepancy δ . Then, for a given combination of nozzle temperature and speed, the predicted model discrepancy is used to correct the bond length estimated by the neck growth model.

A series of experiments with different combinations of nozzle temperature and printer speed are used to measure the bond length, thus providing discrepancy data to train and test the above GP surrogate model. The specimens were sectioned at the midpoint, i.e. $z_{\text{cut}} = L/2 = 0.0175$ m, and their cross-sections were analyzed under a digital microscope. The features of these cross-sections were analyzed using the image processing program ImageJ [88] (Fig. 7.14).

For a sample that is printed with inputs $(T_n, v_p) = (240^\circ\text{C}, 0.042 \text{ m/s})$, the intra-layer bond lengths between adjacent filaments at $z_{\text{cut}} = 0.0175$ m of each layer are given in Table 7.2. The numbering of the interfaces is done from left to right for all layers. For example, the label for the interface between 1st and 2nd filaments is 1 and the label for the interface between 89th and 90th filaments is 14.

Table 7.2: Intra-layer bond length measurements at each layer for $(T_n, v_p) = (240^\circ\text{C}, 0.042 \text{ m/s})$

Layer	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0.4146	0.4558	0.5050	0.5050	0.5130	0.4339	0.4425	0.3915	0.3806	0.3930	0.3840	0.3798	0.3323	0.3162
2	0.4529	0.5056	0.5398	0.5357	0.5345	0.5050	0.3350	0.3293	0.3709	0.3833	0.4041	0.3854	0.2870	0.2606
3	0.4203	0.4828	0.5469	0.5517	0.5046	0.4889	0.3771	0.2269	0.2406	0.3271	0.3284	0.3361	0.2758	0.2506
4	0.4619	0.4895	0.5472	0.5944	0.5917	0.5196	0.3472	0.2055	0.2266	0.2608	0.3069	0.3316	0.2780	0.2781
5	0.4268	0.4798	0.5002	0.6125	0.5746	0.5667	0.4010	0.2617	0.1874	0.3071	0.2743	0.2913	0.2334	0.2621
6	0.4185	0.4694	0.4891	0.5334	0.4948	0.4750	0.3427	0.2617	0.2356	0.2886	0.2771	0.2670	0.2731	0.2884

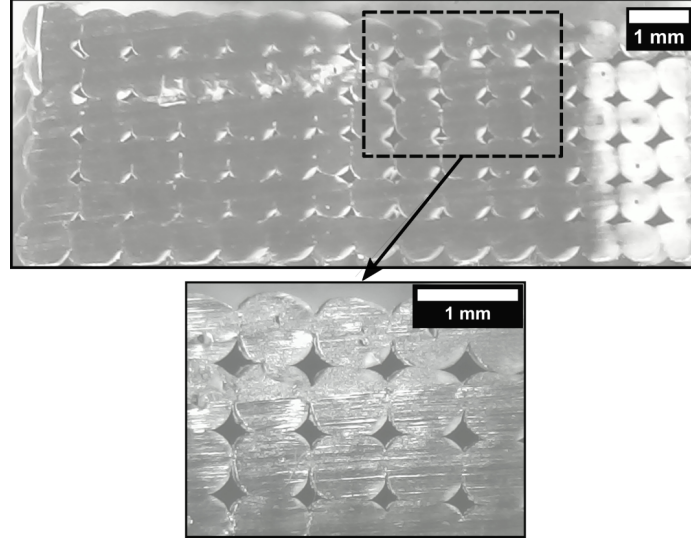


Figure 7.14: Cross-section view of a sample produced with $T_n = 240^\circ\text{C}$, and $v_p = 0.042 \text{ m/s}$

Using Latin hypercube sampling, 25 sets of process parameters were generated, and experiments were conducted at these 25 values. The experimental data is divided into 2 sets, namely training set (20 specimens) and testing set (5 specimens). The training set is further subdivided into two subsets for cross-validation; k -fold cross-validation is performed by splitting the training set into 16 sets for model training and 4 sets for cross-validation, and these sets are selected randomly $k = 5$ different times. The bond length predictions using the observed temperature profiles are different than the ones using the heat transfer model predictions as inputs to the sintering neck growth model. Thus, two different GP models are built to represent the model error associated with these two cases, i.e., the bond length predictions using (A) heat transfer model predictions, and (B) observed temperature data. In each fold of training and cross-validation, the GP models are trained using 16 sets of data and the trained models are cross-validated using the remaining 4 sets of data. The average cross-validation accuracy of the GP models over the 5 folds (random shuffles) is assessed by evaluating the average mean squared error (MSE), which is found to be 0.0018 and 0.0017 for cases A and B respectively.

Further validation of the corrected bond length prediction model is done using testing data, i.e., 5 sets of experiments. The prediction accuracy of the corrected model for the test set is assessed by evaluating the mean squared error (MSE), which is found to be 0.0020 and 0.0019 for cases A and B respectively. MSE of 1% is used as the quantitative criterion for acceptance. The results indicate that the MSE values for both models are less than 1%, thus they are accepted for further analysis. The corrected bond length predictions at each interface based on case A and B are validated by

comparing with the bond length measurements at each interface between adjacent filaments of the test set (see Figs. 7.15 and 7.16. Note that the points are close to the 45-degree line, showing good agreement between predictions and observations even for individual interfaces.

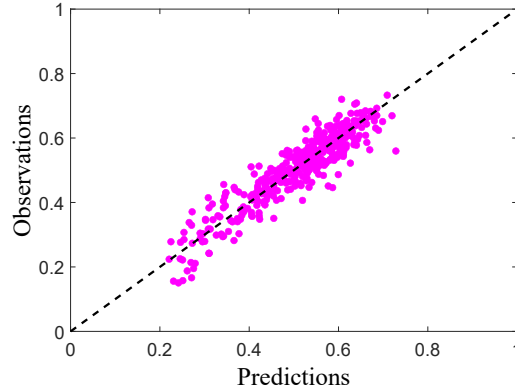


Figure 7.15: Validation of the bond length prediction model for case A, where x and y-axes represent the bond length predictions and experimental observations respectively at each interface between adjacent filaments

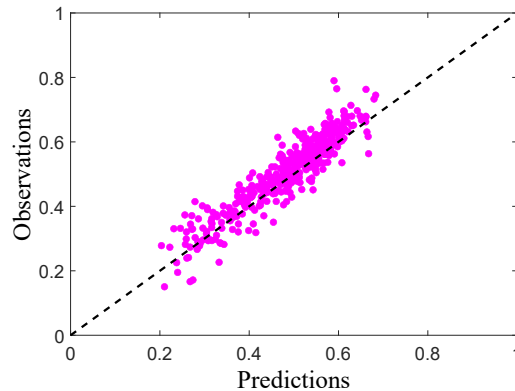


Figure 7.16: Validation of the bond length prediction model for case B, where x and y-axes represent the bond length predictions and experimental observations respectively at each interface between adjacent filaments

7.2.1.4.5 Process Design Optimization

In this optimization, the process parameters (nozzle temperature and extrusion speed) are the design variables. The objective is to optimize the bond quality (indicated by bond length) at each layer of the specimen while satisfying the constraints on the design variables. The lower and upper bounds (LB and UB respectively) for the design variables are shown in Table 7.3. The lower and upper bounds have the same numerical values for all layers. The upper bound for the printer nozzle temperature was chosen as 260°C, since the printer did not allow an extrusion temperature above

260°C. The lower bound for the printer nozzle temperature was chosen as 210°C because the quality of the specimens reduced significantly below a nozzle temperature of 210°C. The lower and upper bounds for the printer extrusion speed v_p were chosen as 0.015 and 0.043 m/s, respectively. We observed debonding and geometrical inaccuracies (warping) in the FFF parts that are printed using process parameter combinations above the linear fit. For example, a part printed with 220°C and 40 mm/s resulted in debonding in several layers and interfaces. These experimental data points and a linear fit to these combinations of process parameters that result in poor bonding (bonding frontier) are plotted in Fig. 7.17. The overall bond quality of a part printed with parameter values above the bonding frontier is poor and delamination is observed. Whereas, the bond quality with parameter values below the bonding frontier is good, and gets better as the distance increases.

Table 7.3: Lower and upper bounds for the process design variables

Design variable	LB	UB
T_n (°C)	210	260
v_p (m/s)	0.015	0.043

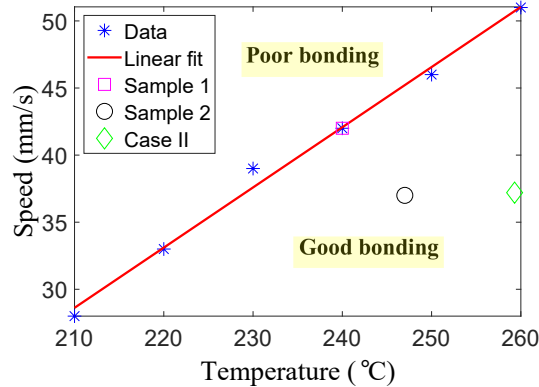


Figure 7.17: Bonding frontier for Ultimaker 2 Extended +

The methodology proposed in Section 7.2.1.2 is implemented here for the part shown in Fig. 7.3a with $L = 0.035$ m. The calibrated values of the material property α and model parameter β in case A are used in the optimization of the neck growth at each layer. In order to demonstrate the robustness of the proposed formulation, two cases are considered: (I) different printer nozzle temperature and extrusion speed values for each layer, and (II) same printer nozzle temperature and extrusion speed value for all layers. The optimal solutions for these two cases are presented in Table 7.4.

The optimal solutions are used to print three specimens for each case. The optimization results are then validated by comparing the bond length model predictions with the bond length measure-

Table 7.4: Optimal printer nozzle temperatures ($^{\circ}\text{C}$) and extrusion velocities (m/s)

Layer	Case I		Case II	
	T_n	v_p	T_n	v_p
1	259.91	0.0430	259.30	0.0372
2	259.65	0.0162	259.30	0.0372
3	258.63	0.0150	259.30	0.0372
4	258.87	0.0150	259.30	0.0372
5	258.99	0.0150	259.30	0.0372
6	258.99	0.0150	259.30	0.0372

ments at each interface between adjacent filaments. The MSE value of 0.0027 is obtained based on the parts printed with the optimal solutions. Figure 7.18 shows the validation results for all six specimens, i.e., three specimens for each of Case I and II, printed with the corresponding optimum process parameter solution. The mean and standard deviation of bond lengths of each layer are

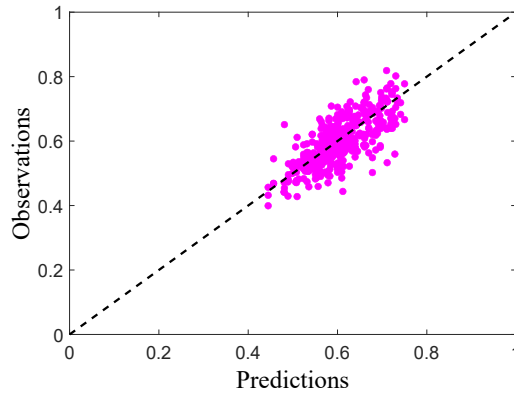


Figure 7.18: Validation of the optimization results, where x and y-axes represent the bond length predictions and experimental observations respectively at each interface between adjacent filaments of Case I and II, respectively (three specimens printed for each case)

averaged across the three specimens ($\bar{\mu}_{\text{BL}}$ and $\bar{\sigma}_{\text{BL}}$). The standard deviation across the averaged mean of bond lengths of three parts σ_{parts} are shown in Table 7.5 together with $\bar{\mu}_{\text{BL}}$ and $\bar{\sigma}_{\text{BL}}$. It is seen in Table 7.5 that the variations across these specimens are relatively small. Note that the averaged mean of bond lengths ($\bar{\mu}_{\text{BL}}$) across these three specimens is found to be smaller at the first layer than the other layers for the first case. The likely reasons for this difference are that the calibration/leveling of the build plate can be erroneous, and/or the printer extrusion speed decreases significantly (from 0.0430 m/s to 0.0162 m/s) as the second layer starts printing. These may result in excessive deformation on the first layer due to gravity and/or the weight of the material deposited above the first layer. As the bond quality starts reaching its upper limit (i.e., dimensionless neck radius $y/a = 1$) as shown in Fig. 7.19c, the differences regarding the bond length between the top

and bottom layers become negligible. Whereas, in the second case, since the dimensionless neck radius is still relatively less than unity, the decrease in the neck growth in the top few layers is more prominent. This difference can be attributed to the fact that the top layer cools down more than the bottom layers because a larger surface area of the filaments in the top layer is exposed to the environmental temperature. This results in relatively poor bonding in the top layers. As it can be seen in Table 7.5, the overall bond quality of the part and the bond quality at each layer are significantly better for the first case since the printer nozzle temperature and extrusion speed are optimized at each layer separately.

Table 7.5: Overall average bond length at optimal solutions (all units are in millimeters)

Layer	Case I			Case II		
	$\bar{\mu}_{BL}$	$\bar{\sigma}_{BL}$	σ_{parts}	$\bar{\mu}_{BL}$	$\bar{\sigma}_{BL}$	σ_{parts}
1	0.56	0.0458	0.0479	0.58	0.0458	0.0049
2	0.65	0.0387	0.0292	0.59	0.0385	0.0053
3	0.65	0.0368	0.0146	0.59	0.0567	0.0022
4	0.64	0.0460	0.0086	0.57	0.0568	0.0149
5	0.66	0.0359	0.0068	0.55	0.0602	0.0129
6	0.65	0.0407	0.0048	0.52	0.0494	0.0081

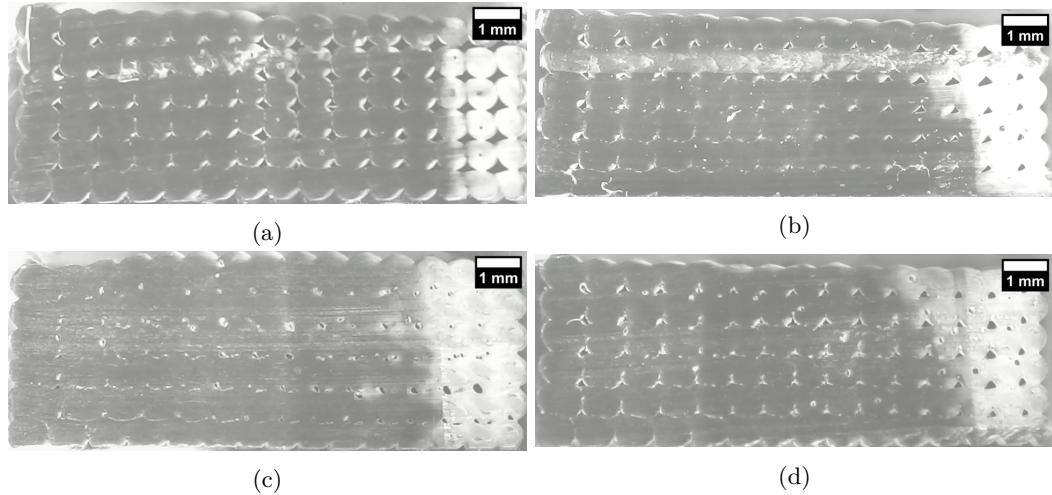


Figure 7.19: Cross-section views of the parts at $z_{cut} = 0.0175$ m built with non-optimal and optimal process parameters: (a) sample 1, (b) sample 2, (c) case I and (d) case II

Two specimens (sample 1 ($T_n = 240^\circ\text{C}$ and $v_p = 0.042$ m/s) and sample 2 ($T_n = 247^\circ\text{C}$ and $v_p = 0.037$ m/s)) that are printed with non-optimal process parameters are used to demonstrate the effect of the proposed methodology. The total void area and the overall mean bond length (BL) at $z_{cut} = 0.0175$ of case I and II are compared with sample 1 and sample 2 in Table 7.6. The cross-section views of these parts are shown in Fig. 7.19. The overall bond quality of sample 2 represented by

total void area (0.54 mm^2) and overall mean bond length metrics (0.52 mm) is better than the bond quality of sample 1 since the total void area and overall mean bond length of sample 1 are 2.11 mm^2 and 0.39 mm respectively (lower values of total void area and higher values of overall mean bond length imply a better-quality product). The total void area of the parts is identified with the use of microscopy images processed through the ImageJ software [88], combined with a Matlab script to estimate the size of voids. The total void area is the smallest and the overall mean bond length is the largest for case I as expected, thus demonstrating the effectiveness of the proposed optimization methodology.

Table 7.6: Total void areas and overall mean bond lengths at $z_{\text{cut}} = 0.0175 \text{ m}$

Metric	Case I	Case II	Sample 1	Sample 2
Total void area (mm^2)	0.42	0.47	2.11	0.54
Overall mean BL (mm)	0.64	0.57	0.39	0.52

The coupled heat transfer and sintering neck growth model is directly used in the uncertainty quantification and optimization problems since the original simulation model is not very expensive ($\sim 100 \text{ s}$ for one run on Intel® Xeon® CPU E5-2650 v4@2.20GHz with 64 GB RAM desktop machine).

7.2.1.5 Summary

This section developed a formulation for FFF process optimization under uncertainty, using an analytical solution for the transient heat transfer during filament deposition and cooling, and a sintering neck growth model. The neck growth between adjacent filaments is optimized while accounting for various sources of uncertainty and error. Variance-based sensitivity analysis is used to quantify the contribution of each uncertainty source to the variability of the output quantity (bond length). The physics model parameters that have the most significant contribution to the uncertainty in the model output are calibrated using experimental measurement of bond length. Additional experimental observations are used to build a surrogate model for the physics model discrepancy in predicting the bond length. The surrogate model is used to estimate the model discrepancy for given process parameter values and correct the physics model predictions. The corrected prediction model is used to select the optimal process parameters to maximize the bond quality at each layer. The optimum solution is validated with specimens printed with the optimized process parameters.

7.3 Multiple Quantities of Interest^{3,4}

7.3.1 Proposed Methodology

In this section, we present the methodology for process parameter optimization under uncertainty with a focus on the FFF process. However, the proposed methodology is applicable to any AM process with corresponding data and QoI prediction models. The three components of the methodology are: (a) Data collection, (b) Construction of data-driven prediction models including uncertainty, and (c) Multi-objective optimization under uncertainty.

7.3.1.1 Data Collection

For a data-driven methodology, the first step is to collect data to build the prediction models for the QoI. Traditionally, data is collected from physical experiments. With the advent of physics-based modeling, some researchers also use data generated by the physics-based model [177]. In that case, the physics-based model prediction should be first validated with experiments to ensure that the training data closely approximates the actual physics of the AM process. In some cases, data available from previous studies or in the public domain might also be used [38, 178–180]. For this work, we collected data from laboratory experiments to build the prediction models. The shape of the part is conceptualized, and a CAD model is first built and then sliced in a slicing software where the printing path is also defined. The printing instructions thus generated are used to print the part.

Various sensors can be used to monitor the AM process and parts. The most common measurement QoIs are melt pool temperature, part dimension, surface roughness, microstructure, tensile properties, etc. Monitoring can be broadly divided into online and offline monitoring techniques. Online monitoring refers to in situ monitoring of the part during the manufacturing process in order to implement process control. Effective online monitoring is not disruptive to the ongoing process, and is non-destructive to the part such as monitoring of the temperature profile using an infrared (IR) camera, or monitoring of the part geometry using profilometer or optical camera. Offline or ex situ monitoring of the part after it has been produced may be either destructive (such as microstructure characterization with scanning electron microscope) or non-destructive (such as part dimension measurement using calipers). Both contact and non-contact techniques may be used for monitoring. For example, temperature data can be collected through contact thermocouples or non-contact IR cameras. Thus, depending on the QoI and the monitoring mode, the instruments and experimental setup are determined. The collected data is then used to construct the prediction models. In this

³Adapted with permission from: [Kapusuzoglu, B., Nath, P., Sato, M., Mahadevan, S., & Witherell, P., “Multi-Objective Optimization Under Uncertainty of Part Quality in Fused Filament Fabrication,” ASME J. Risk Uncertainty Part B., vol. 8\(1\), no. 011112, 2022.](#)

⁴© 2022 by ASME

paper, the QoIs are part thickness and bond quality, both of which are measured offline after the part is manufactured.

7.3.1.2 Construction of Data-Driven Prediction Model

There are several machine learning techniques to build prediction models using data. Commonly used approaches include Decision Tree, K-Nearest Neighbor, Support Vector Machine, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Artificial Neural Network (ANN), Polynomial Chaos Expansion, Radial Basis Function, Gaussian Process Modeling, Random Forest Regression etc. As the amount of data available has increased, use of ANN has gained popularity; several types of ANN are available, such as deep neural network (DNN), recurrent neural network (RNN) to handle time series data, and convolution neural network (CNN) to handle image data. Machine learning (ML) models appear promising for complex systems that are not fully understood or cannot be represented with simplified physics-based relationships, given adequate quality and quantity of data. Generally, the construction of data-driven ML models does not require in-depth knowledge of the complex physics inherent in the physical process [104]. In this work, adequate experimental data is available to build a deep learning (DL) model based on the observation data, thus we pursue the ML approach. In addition, we quantify the DL model uncertainty by constructing Bayesian neural network (BNN) models (see Section 2.2.3) to predict the QoIs and use these BNN models to perform optimization under uncertainty. In the next sections, we introduce the underlying mechanisms in the feedforward neural network and Bayesian neural network (BNN).

7.3.1.2.1 Uncertainty Quantification in Bayesian Neural Network (BNN)

Various sources of uncertainty can be considered, such as (a) Epistemic uncertainty due to lack of knowledge, and (b) Aleatory uncertainty due to the inherent variability across multiple samples and over space and time. Epistemic uncertainty is caused by insufficient knowledge or information about the model and data. In the case of a DNN model, the values of the model parameters such as neuron weights are estimated from the data. If limited data is available, there is epistemic uncertainty in the model prediction. BNN is used to describe the epistemic uncertainty caused by the model by placing distributions over the network weights.

Aleatory uncertainty is caused by the natural variability in the AM process, leading to variability in the process output QoI. For example, the AM parts printed with the same process parameters may have different geometric dimensions and mechanical properties. The observation noise parameter σ also needs to be tuned. Similar to the procedure in Bayesian model calibration, observation noise

can be learned through the BNN model by minimizing the following loss function [181]:

$$\mathcal{L}_{BNN}(\boldsymbol{\theta}) = \frac{1}{N} \sum_i \frac{1}{2} \hat{\sigma}_i^{-2} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 + \frac{1}{2} \log \hat{\sigma}_i^{-2} + \lambda g(\mathbf{w}) \quad (7.7)$$

where N is the number of observations \mathbf{y}_i ; $\hat{\mathbf{y}}_i$ and $\hat{\sigma}_i^2$ are the predictive mean and observation noise corresponding to input indexed by i , the first term represents the norm of residuals for measuring goodness of fit; λ is the weight decay parameter, and $g(\mathbf{w})$ represents a regularization function for the weights (or the penalty term which often uses L_2 regularization). The second regularization term prevents the network from predicting infinite uncertainty, thus zero loss. The neuron weights can be drawn from the approximate posterior $\hat{\mathbf{w}} \sim q(\mathbf{w})$ to obtain the model outputs of a BNN denoted as $\mathbf{f}^{\hat{\mathbf{w}}}(\mathbf{x}) = [\hat{\mathbf{y}}, \hat{\sigma}^2]$. The second term of the loss function can be regarded as an uncertainty regularization term and it prevents the network from predicting infinite uncertainty. In order to have a numerically stable network during training, the log variance $\log \hat{\sigma}_i^2$ is predicted instead of predicting $\hat{\sigma}_i^2$. (Sometimes the process parameter settings specified by the designer (such as the nozzle temperature, nozzle speed, layer thickness, etc.) may not be actually realized in manufacturing (this is input uncertainty (epistemic), i.e., the input value specified in the model is different from what is actually in the manufacturing process, and we do not know what the actual value is). However, this type of uncertainty is not considered in this paper).

The predictive mean and variance are estimated by collecting the results of stochastic forward passes through the model. The mean prediction of the model with T MC samples can be approximated by

$$\mathbb{E}(\mathbf{y}) \approx \frac{1}{T} \sum_{t=1}^T \hat{\mathbf{y}}_t, \quad (7.8)$$

and the variance of the prediction is estimated by

$$\text{Var}(\mathbf{y}) \approx \underbrace{\frac{1}{T} \sum_{t=1}^T \hat{\mathbf{y}}_t^2 - \left(\frac{1}{T} \sum_{t=1}^T \hat{\mathbf{y}}_t \right)^2}_{\substack{\text{model uncertainty} \\ \text{(epistemic)}}} + \underbrace{\frac{1}{T} \sum_{t=1}^T \hat{\sigma}_t^2}_{\substack{\text{observation} \\ \text{noise} \\ \text{(aleatory)}}}. \quad (7.9)$$

In the next section we discuss how to use the model predictions given by Eq. 7.8 and Eq. 7.9 for optimization under uncertainty.

7.3.1.3 Multi-Objective Optimization under Uncertainty

7.3.1.3.1 Formulation of Multi-Objective Optimization

The objective functions in Eq. 7.1 depend on the intended use of the additively manufactured part. Since optimization involves evaluation of the objective function repeatedly at different process parameter settings, instead of conducting expensive physical experiments the objective functions are evaluated using the data-driven model discussed in Section 7.3.1.2. The prediction from the BNN model has a mean given by Eq. 7.8 and the corresponding variance given by Eq. 7.9. The conventional deterministic optimization does not consider the uncertainty in the input variables, data and model parameters; the output of the manufacturing process is sensitive to these uncertainty sources. Optimization under uncertainty can be pursued in two directions: (1) robust design optimization (RDO) [163], and (2) reliability-based design optimization (RBDO) [164]. In RDO, both the mean and the variability of the objective function are optimized (since minimizing the variability makes the objective insensitive to variations of the input variables and parameters), and the constraints are satisfied within specified uncertainty bounds. On the other hand, in RBDO, a desired target level of reliability is maximized (i.e., probability of satisfying a desired threshold of performance or quality) by optimizing the decision variables, or a cost function is minimized while satisfying a reliability constraint. In this work we are interested in a robust design to improve the quality of products and processes by optimizing both the mean and variance of the quantities of interest. Therefore, robust design optimization (RDO) is used in this study for the design of the FFF process parameters. (Note also that for well-designed practical systems, the probability of failure would be very low; thus the RBDO formulation would require substantially more function evaluations in comparison to the RDO formulation where the means and variances of the objective function and the constraints can be evaluated with a much smaller number of function evaluations). The robustness of the objective function can be achieved by simultaneously optimizing the mean and variance. Thus RDO even with respect to a single objective QoI becomes a bi-objective optimization problem with two objectives: (a) Optimize the mean of the QoI, and (b) Minimize the variance of the QoI. The resulting bi-objective RDO problem can be approximately solved through a single objective formulation, using a weighted sum approach, i.e., mean and variance terms have weights that reflect the designer's preference.

The computation of objectives $f_j(\mathbf{x}, \mathbf{p})$ ($j = 1, \dots, n_{obj}$) and constraints $\mathbf{g}(\mathbf{x}, \mathbf{p})$ is affected by uncertainty sources such as input variability and model uncertainty; thus we have stochastic objectives and constraints. The input variability is already present in the experimental data that is used for

training the DL model, and further, the model uncertainty (epistemic) is also captured through the BNN approach. The aleatory uncertainty is learned by minimizing the loss function shown in Eq. 7.7. MC dropout, which is performing dropout during prediction, is used to quantify the epistemic uncertainty in the model prediction. As a result, the output of the BNN model is stochastic, which incorporates both sources of uncertainty mentioned above. The optimization formulation in Eq. 7.2 properly accounts for the resulting uncertainty. In the next section, we discuss the solution of the optimization problem.

7.3.1.3.2 Solution of Multi-Objective Optimization

Depending on the design variables and constraints, the optimization problem discussed in Section 7.3.1.3 can be solved using a suitable global optimization algorithm. In this paper, we choose to employ Non-dominated Sorting Genetic Algorithm II (NSGA-II) [168].

7.3.1.3.2.1 Non-Dominated Sorting Genetic Algorithm II

Since a multi-objective problem is formulated, the solution depends on the weighting coefficients. The trade-off among n_{obj} objective functions f_k , $k = 1, 2, \dots, n_{obj}$ can be represented by the Pareto front. The Pareto front is a set of Pareto optimal solutions, which correspond to the solutions for different values of the weighting coefficients for the optimization problem specified in Eq. 7.2. The NSGA-II algorithm is one of the frequently applied multi-objective optimization evolutionary algorithms for generating the Pareto front [168].

The NSGA-II procedure for finding the Pareto front can be briefly described in following steps:

1. An initial random population is generated and the fitnesses of the individuals are evaluated during several generations;
2. Several Pareto fronts are generated by ranking the population based on the non-dominating sorting criteria (where individuals with the best rank represents the first front, the ones with the second best rank generate the second front and so on);
3. The crowding distance value is assigned to each front once the sorting is completed;
4. The individuals are selected using a binary tournament selection with crowded-comparison operator (if two individuals have the same rank, the individual with greater crowding distance is selected to increase the diversity, otherwise an individual with a better rank is chosen);
5. Binary crossover and polynomial mutation are used to generate a new offspring population combined with the current population;

6. The next generation is set by selection until the population size exceeds the current population; and
7. The above steps are repeated until the stopping condition is met and a set of non-dominated Pareto optimal solutions are obtained.

The Pareto optimal set is the set of all possible Pareto optimal vectors and there is a need for a stopping criterion that evaluates the quality of the Pareto front solutions. There are methods that apply a stopping criterion, such as NSGA-II [168]. Another important concept is the ideal stopping generation criterion, where the solution is as close to the optimal Pareto front as possible in a short amount of generation. The ideal stopping generation is a compromise between the distance to the Pareto optimal front and the cost of generation. Therefore, it is important to be able to determine how good a solution is compared to the optimal one by using an indicator, such as hypervolume indicator [182], and epsilon indicator [183, 184].

7.3.1.3.2.2 Evaluating Performance: Hypervolume Indicator

Among the variety of performance indicators for genetic algorithm in optimization, the hypervolume indicator has been favored by many researchers to measure the quality of a solution [182, 185]. The hypervolume indicator can capture the closeness of the solutions to the Pareto optimal set and partially the spread of the solutions across the objective space.

The hypervolume indicator, $\mathcal{I}_{hv}(\mathcal{A})$, computes the volume of the region, H , defined by a set of reference or nadir points and given set of points, \mathcal{N} and \mathcal{A} , respectively as

$$\mathcal{I}_{hv}(\mathcal{A}) = \text{Volume} \left(\bigcup_{\forall a \in \mathcal{A}; \forall n \in \mathcal{N}} \text{hypercube}(a, n) \right), \quad (7.10)$$

where larger values of $\mathcal{I}_{hv}(\mathcal{A})$ corresponds to better solutions. The absolute performance of an optimization algorithm is measured using nadir points, which are the worst elements of the Pareto front solutions.

The performance of an algorithm as the evolution proceeds can be tracked by transforming the indicator [183]:

$$\mathcal{I}_{hv}(t) = \mathcal{I}_{hv}(\mathcal{P}_t) - \mathcal{I}_{hv}(\mathcal{P}_{t-1}), \quad (7.11)$$

where \mathcal{P}_{t-1} and \mathcal{P}_t are the previous and current non-dominated elements of the local Pareto optimal front.

7.3.1.3.3 Experimental Validation

The subsection discusses two types of validation as part of the methodology: first the prediction model is validated against experimental data, and second, the optimization solution is also validated against experimental data. (Note that there is also cross-validation during the model construction phase, by partitioning the data into training and testing sets, in order to check that the model has the required accuracy, as explained in Section 7.3.2.3).

7.3.1.3.3.1 BNN Model Validation

Quantitative comparison of model prediction against experimental data is traditionally done using hypothesis testing. In recent model validation literature, several quantitative methods have been pursued, such as classical hypothesis testing, Bayesian hypothesis testing, reliability-based method, area-metric-based method, etc. Any of these methods can be used to quantitatively validate the optimization results [186]. The classical t -test is used in this paper to test the null hypothesis that the mean of observations is equal to the mean of the model prediction. The t -test is based on Student's t -distribution and the corresponding test statistic t is

$$t = \frac{\bar{Y}_D - \mu_m}{s_D / \sqrt{n}} \quad (7.12)$$

where \bar{Y}_D is the sample mean of experimental observations, μ_m is the mean prediction, and s_D is the sample standard deviation. The p -value (i.e., $p = 2F_{T,n-1}(-|t|)$, where $F_{T,n-1}$ is the cumulative distribution function (CDF) of a t -distribution with $(n-1)$ degrees of freedom) is compared with the significance level α (usually 0.01 or 0.05). If the p -value is less than or equal to α , then the null hypothesis is rejected, otherwise the null hypothesis is not rejected.

7.3.1.3.3.2 Optimization Solution Validation

Since we are using an approximate surrogate model, we perform the second validation to check whether the propagation of surrogate model error through the optimization process has significantly affected the optimum solution. Validation of the optimization methodology can be achieved by comparing the performance of the optimum solution against other randomly selected settings for the decision variables. In this paper, the Pareto optimal solutions obtained using the proposed methodology are validated by conducting actual printing experiments at the optimal process parameter settings and other random settings. The performance comparison is done by comparing the quality objectives achieved in the actual manufactured parts, such as mean bond length, mean

part thickness, and their standard deviations. Section 7.3.2.4 considers four combinations of these individual objectives. In each case, it is investigated whether the optimum solution gives better part quality than randomly selected process parameter settings.

7.3.1.4 Summary of Methodology

The steps of the proposed methodology for multi-objective optimization of AM process parameters under uncertainty are summarized in Fig. 7.20. The proposed method consists of five main components: (a) Collection of experimental data, (b) Construction of data-driven probabilistic prediction models, (c) Validation of the trained BNN models, (d) Multi-objective optimization under uncertainty of process parameters, and (e) Validation of the optimization results.

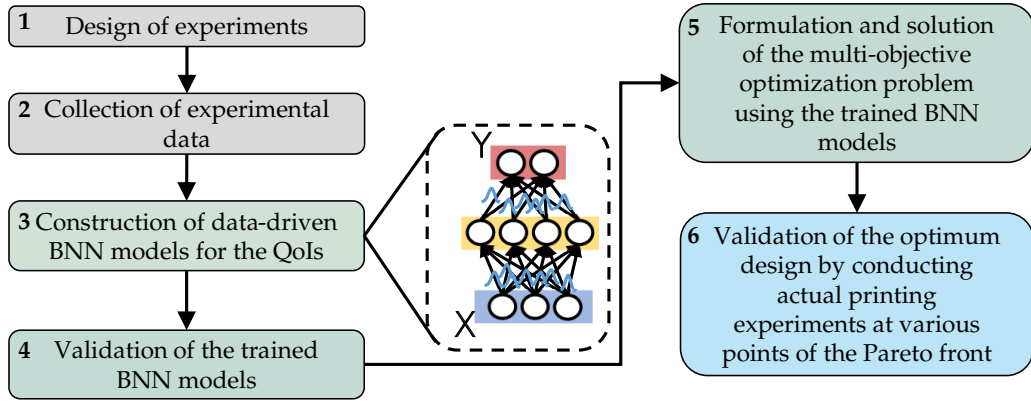


Figure 7.20: Flowchart of the proposed methodology

The methodology presented in this paper can be generalized for any AM process, as long as experimental data is available to build the data-driven model. In the next section, we demonstrate the effectiveness of the proposed methodology for four different cases with multiple objectives.

7.3.2 Numerical Example

In this section we demonstrate the implementation of the proposed methodology on a part of dimensions $35 \text{ mm} \times 12 \text{ mm} \times 4.2 \text{ mm}$ printed with ABS (acrylonitrile butadiene styrene) using an Ultimaker S5 printer [187]. The objective is to find the optimal process parameters nozzle temperature (T_e), nozzle speed (V_e), and layer height (l_t), i.e., $x = [T_e, V_e, l_t]$ such that both the dimensional accuracy and bond quality of the part are maximized. Since the maximum extrusion volume for the 0.8 mm diameter nozzle used in the experiments is $24 \text{ mm}^3/\text{s}$, an inequality constraint $g = V_e \times l_t \times L_w \leq 24 \text{ mm}^3/\text{s}$, where $L_w = 0.8 \text{ mm}$ is the raster width, is added to the optimization problem.

7.3.2.1 Data Collection from Experiments

In this work, we use ABS and modify the printing environment by adding an enclosure to the 3D printer to isolate the printer from environmental effects (see [26, 27, 32, 104] for details). Using Latin hypercube sampling (Fig. 7.22), 25 sets of process parameters x are generated and three parts are printed at each parameter setting. The ranges considered for the variables are: T_e : 215°C to 280°C, V_e : 25 mm/s to 45 mm/s, and $l_t = \{0.42, 0.60, 0.70\}$ mm.

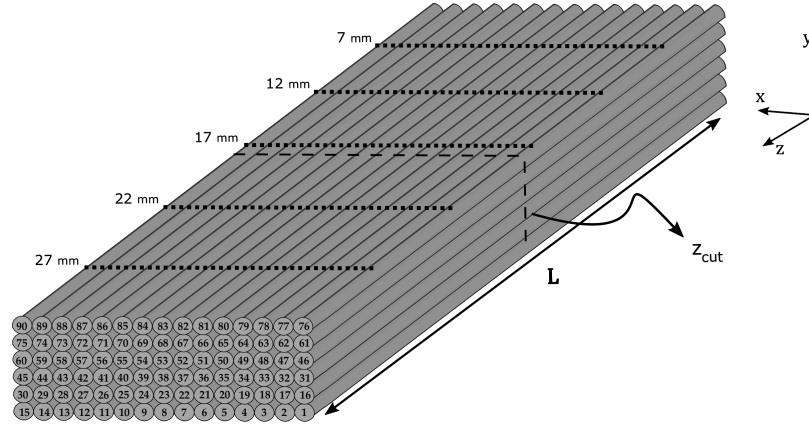


Figure 7.21: Deposition sequence of unidirectional filaments

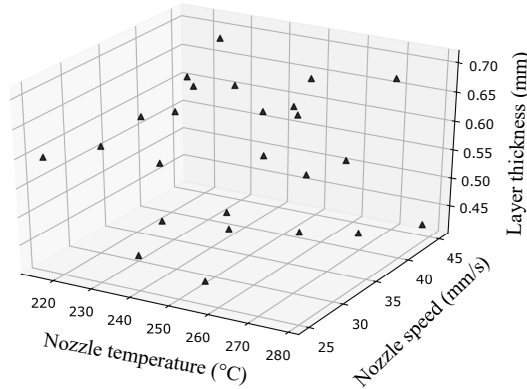


Figure 7.22: Design of experiments for process parameters

Since the focus is on maximizing the dimensional accuracy and bond quality, data pertaining to these two quality characteristics are collected. The measure for dimensional accuracy is considered to be the error in part thickness, i.e., the difference between the printed part thickness and the target part thickness of 4.2 mm. As shown in Fig. 7.23, using a laser displacement sensor Keyence LK-H057 [188] the part thickness is measured at $z = \{7, 12, 17, 22, 27\}$ mm at discrete points along the x-axis. The bond lengths (mesostructural feature of interest) between the filaments were measured at cross-section $z_{cut} = L/2 = 17.5$ mm as shown in Fig. 7.21 with the use of microscopy images processed

through the ImageJ software [88].

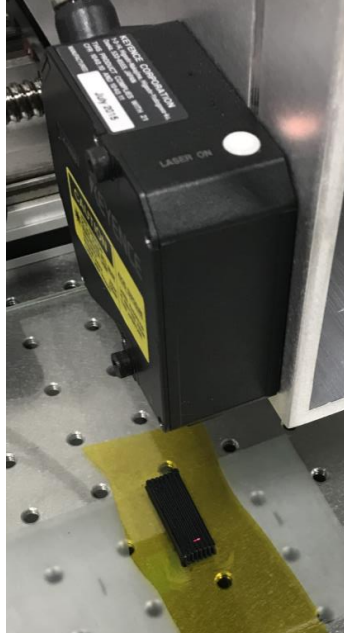


Figure 7.23: Thickness measurement of the printed part

The bond length measurements at each interface of a part, which is printed with inputs $(T_e, V_e, l_t) = (227^\circ\text{C}, 41 \text{ mm/s}, 0.6 \text{ mm})$, are given in Table. 7.7. The interfaces are numbered from left to right for all layers (e.g., the label for the interface between filaments 1 and 2 is 1 and the label for the interface between filaments 89 and 90 is 14 in Fig. 7.21). The bonding quality between adjacent

Table 7.7: Bond length measurements (in mm) at each layer for $(T_e, V_e, l_t) = (227^\circ\text{C}, 41 \text{ mm/s}, 0.6 \text{ mm})$

Layer	Interface													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0.383	0.381	0.416	0.469	0.459	0.487	0.472	0.490	0.528	0.525	0.500	0.474	0.452	0.421
2	0.431	0.444	0.452	0.429	0.396	0.431	0.322	0.360	0.294	0.353	0.363	0.408	0.317	0.342
3	0.462	0.495	0.434	0.487	0.431	0.345	0.000	0.167	0.101	0.180	0.281	0.347	0.251	0.220
4	0.424	0.464	0.416	0.444	0.439	0.314	0.170	0.000	0.220	0.261	0.248	0.238	0.210	0.185
5	0.365	0.441	0.419	0.441	0.431	0.281	0.162	0.000	0.246	0.266	0.243	0.177	0.233	0.200
6	0.375	0.398	0.467	0.510	0.396	0.347	0.259	0.000	0.254	0.215	0.208	0.157	0.261	0.208
7	0.340	0.381	0.480	0.449	0.391	0.360	0.287	0.223	0.180	0.223	0.134	0.182	0.195	0.193

filaments at $z_{\text{cut}} = 17.5 \text{ mm}$ of the same part is illustrated in Fig. 7.24. The 8th and 9th filaments at layers 3, 4, 5, and 6 show delamination. This delamination phenomenon was also observed at the exact same interfaces in the other two sets of samples printed with the same process parameter values. The average of these measured bond lengths gives the mean bond length of the part.

The thickness of the part printed at $(T_e, V_e, l_t) = (227^\circ\text{C}, 41 \text{ mm/s}, 0.6 \text{ mm})$, measured using the

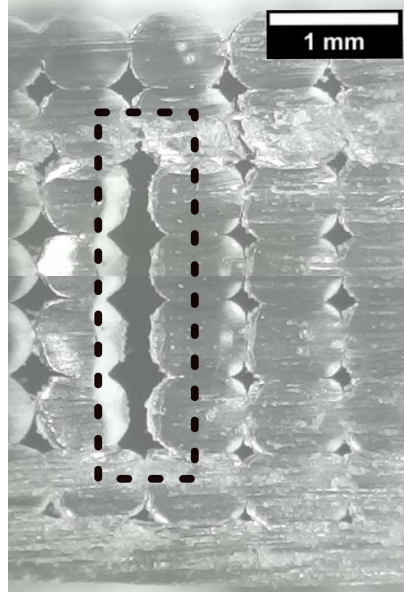


Figure 7.24: Cross-section view of a sample with poor bonding

laser sensor, is shown in Fig. 7.25. At each of the five Z-axis points, the measurements are taken along the X-axis to measure the part thickness. It is observed that repeated measurements along X-axis for each Z-axis point show similar values demonstrating the reliability of the measuring system. These discrete point measurements averaged together give the mean thickness of the part.

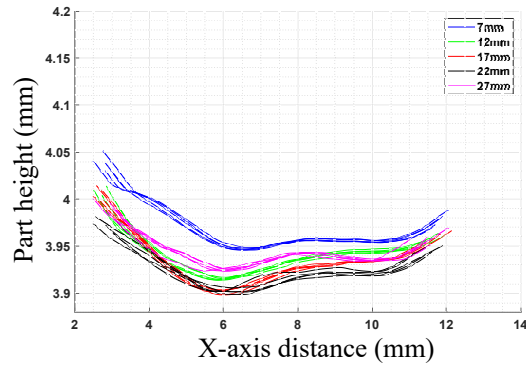


Figure 7.25: Part thickness of a sample along Z-axis locations

7.3.2.2 Model Training and Prediction

Separate BNNs are constructed for predicting each individual QoI (bond length and thickness in this case). Since the data is collected at many discrete spatial locations, the inputs to the BNNs include the process parameters nozzle temperature, nozzle speed, layer height, and also the spatial locations. The inputs to the bond length model (BNN_{bl}) are $[T_e, V_e, l_t, x, y]$ and the inputs to the part

thickness model (BNN_{th}) are $[T_e, V_e, l_t, x, z]$. The output for BNN_{bl} and BNN_{th} are bond length and part thickness respectively.

The available experimental data related to the bond length model and part thickness model is divided into 2 sets, namely training set (75% of all data) and testing set (25% of all data). The input and output data are normalized prior to the training of the BNN models, and the hyperparameters of these models (e.g., batch size, learning rate, dropout rate etc.) are tuned with grid search. The minimum prediction error during prediction for the bond length model, BNN_{bl} , is achieved using a dropout rate of 0.1 with 128 batch size, rectified Linear Unit (ReLU) activation function and Adam optimizer with a learning rate of 0.004. The minimum prediction error of BNN_{th} is obtained using a dropout rate of 0.1 with 128 batch size, sigmoid activation function and Adam optimizer with a learning rate of 0.005.

7.3.2.3 Model Performance and Cross-Validation

In order to measure the accuracy of the trained bond length BNN model (BNN_{bl}), the model predictions for the testing data subset (i.e., subset of 25 percent of all shuffled data set) are compared with the observed bond lengths at randomly selected interfaces of each printed part in Fig. 7.26. The horizontal axis represents test combinations with different input parameter combinations and the blue dots denote the bond length measurements at the randomly selected interface from the test data for each printed part. In this figure, the black cross denotes the mean prediction of bond length as predicted by the probabilistic model, the shaded light brown area demonstrates the epistemic uncertainty for one standard deviation away from the mean value, the shaded cyan represents the aleatory uncertainty for one standard deviation away from the mean plus the epistemic uncertainty value and the shaded light blue represents the total uncertainty (aleatoric and epistemic) for one standard deviation away from the mean plus the epistemic plus the aleatory uncertainty value. Thus, the whole shaded area represents the uncertainty bounds for three standard deviations away from the mean predictions.

Figure 7.27 compares the observed average bond lengths with the predictions using BNN_{bl} . The horizontal axis represents different sets of samples, which are printed using the same process parameters (T_e, V_e, l_t) . An overall bond quality metric is obtained by averaging the measured and predicted bond lengths of a part at each interface. As shown in Fig. 7.27, the model is able to capture the ground truth within half standard deviation.

The observed and mean prediction bond length values at the interfaces of the test data using MC dropout are compared on a 45-degree angle line in Fig. 7.28. The bond length measurements

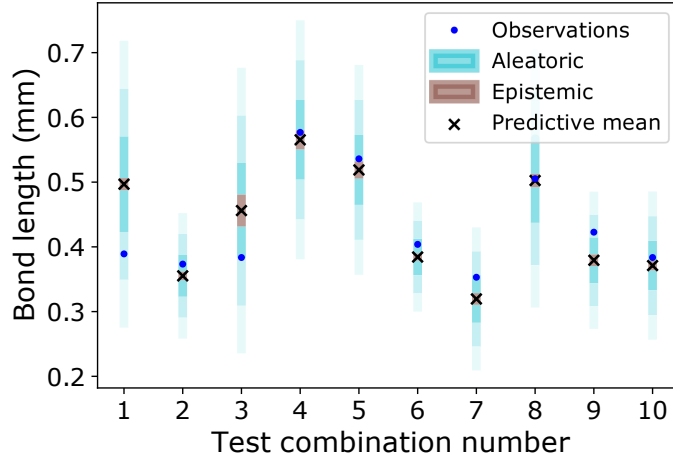


Figure 7.26: Predicted bond lengths and uncertainty bounds at randomly selected interfaces from the test data and actual bond length measurements (obtained using microscopy images)

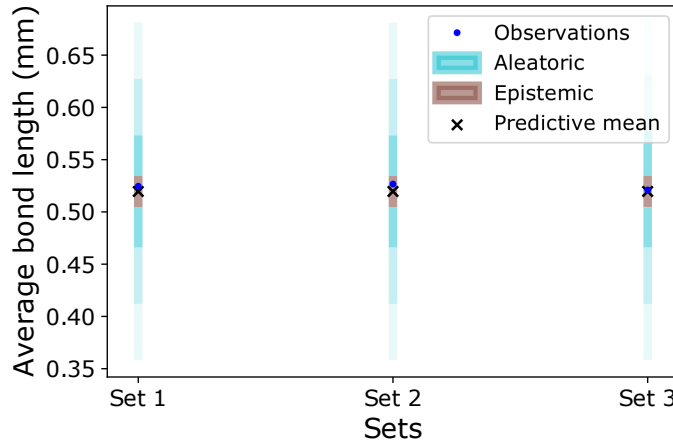


Figure 7.27: Average bond length predictions vs. observations of three sets of parts printed with same process parameters

and predictions are carried out at each interface in a layer. The delamination (i.e., zero bond length, which is an outlier) observed in one of the parts is predicted with a non-zero mean bond length. Similar analysis is carried out for part height measurements. Figure 7.29 shows the comparison between the observation and mean prediction using MC dropout for the part height. Note that the pairs of the observations and the mean predictions are close to the 45-degree line, showing good agreement between predictions and observations for the BNN_{bl} model but not that good for the BNN_{th} model. Further cross-validation of the bond length and part thickness prediction models is done using testing data, i.e., (25% of all shuffled data). The prediction accuracy of both models for the test set is assessed by evaluating the root mean squared error (RMSE); $RMSE = \sqrt{\sum_{i=1}^N (y_i - \hat{y}_i)^2 / N}$, which

is found to be 0.0667 and 0.0826 for BNN_{bl} and BNN_{th} , respectively. RMSE of 0.1 is used as the quantitative criterion for acceptance. The results indicate that the RMSE values for both models are less than 0.1, thus they are accepted for further analysis.

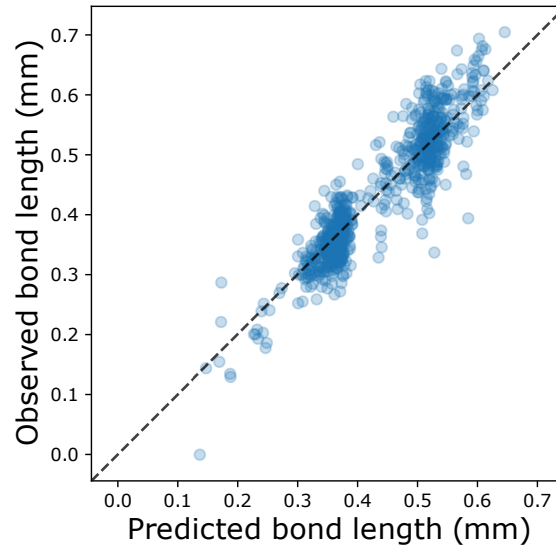


Figure 7.28: Comparison of mean dropout prediction and observation of bond lengths

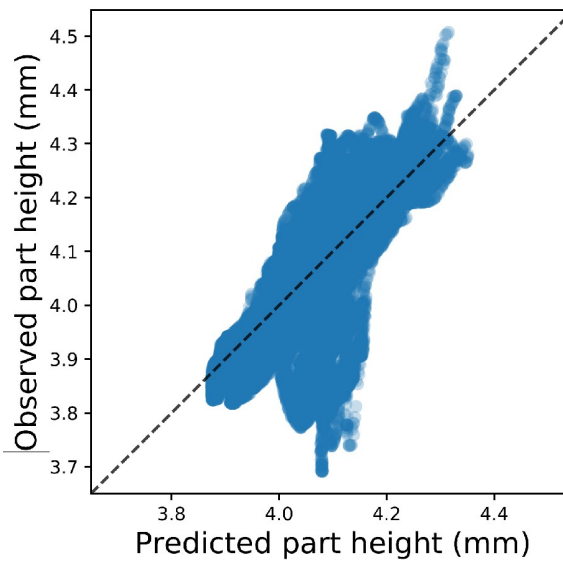


Figure 7.29: Comparison of mean dropout prediction and observation of part thickness

7.3.2.4 Multi-Objective Optimization

The predictions from the BNN models are used in the multi-objective optimization methodology to optimize the process parameters such that the dimensional accuracy and bond quality of the part

are maximized. Four different optimization cases under uncertainty are considered:

- (1) Case 1, the mean overall bond quality of the part is maximized while minimizing its variance;
- (2) Case 2, the mean geometric accuracy of the part is maximized while minimizing its variance;
- (3) Case 3, the mean overall bond quality and the mean geometric accuracy of the part are maximized; and
- (4) Case 4, the mean overall bond quality and the mean geometric accuracy of the parts are maximized while minimizing the variance in bond quality and geometric accuracy.

Note that cases 1, 2, and 4 include minimization of variance of the part quality (bond quality and geometric accuracy), thus considering uncertainties. Case 3 minimizes the means of two part quality metrics without considering the variance. The weighted sum approach gives a convex combination of two different objectives in the first three cases. Whereas, Case 4 is a convex combination of four different objectives (i.e., one minus the mean value of dimensionless bond length of the part, $(1 - \mu_{\hat{b}_l})$, the standard deviation of dimensionless bond length of the part $\sigma_{\hat{b}_l}$, the dimensionless mean and standard deviation of the absolute error between the desired and predicted part thickness $(\mu_{\hat{t}_h}, \sigma_{\hat{t}_h})$).

The QoIs (bond length and part thickness) are dependent on the part layer thickness. For example, intra-layer bond length between interfaces is a function of part layer thickness. The maximum achievable intra-layer bond length equals to layer thickness, thus the scale of bond length values changes with changing layer thicknesses. In order to remove this dependency and to prevent the problem caused by different scales, the mean and standard deviation of the QoIs are non-dimensionalized. The mean and standard deviation of the bond length are non-dimensionalized and scaled to unity by dividing with part layer thickness (i.e., nondimensional bond length = (intra-layer bond length)/(layer thickness)). The predicted mean absolute error in part thickness ($\mu_{\text{th}} = |\mu_{\text{th}_{\text{pred}}} - \text{th}_{\text{desired}}|$, where $\mu_{\text{th}_{\text{pred}}}$ and $\text{th}_{\text{desired}}$ are the mean dropout prediction of part thickness and desired part thickness, respectively) and standard deviation of the part thickness σ_{th} are also non-dimensionalized and scaled to unity as $\mu_{\hat{t}_h} = \mu_{\text{th}}/\text{th}_{\text{desired}}$ and $\sigma_{\hat{t}_h} = \sigma_{\text{th}}/\text{th}_{\text{desired}}$.

Based on the information obtained from the layer height measurements and the infrared (IR) thermal camera images during the printing process, no significant difference is observed between the measured and reference values; thus, the epistemic uncertainty in the input is not considered. The model uncertainty (epistemic) is described by placing distributions over the model’s weights. The aleatory variability (intrinsic randomness) in the input, which can be described as noise in the observations, is present in the experimental data used in training the BNN models and is learned

by minimizing the loss function shown in Eq. 7.7. As a result, uncertainty in the prediction model as well as the aleatory uncertainty in the input are considered in the optimization.

The hyperparameters of the NSGA-II optimization algorithm (i.e., population size and the maximum number of generations) are tuned by evaluating the performance indicator (hypervolume) (see Section 7.3.1.3.2.2) of nine different experiments. The experiments are repeated 30 times since the methods are stochastic. The convergence of hypervolume values are monitored for each case. The probability of crossover and mutation are chosen as 0.8 and 0.2, respectively. For illustration purposes, the hypervolume values for Case 3 are compared by changing the population size or maximum number of generations while keeping the other one constant (Fig. 7.30). The maximum hypervolume value is achieved using a population size of 200 with 30 generations as shown in Fig. 7.30. The hypervolume values of other cases show a similar trend. Fig. 7.31 shows in more detail the change in hypervolume values with increasing numbers of generations for a population size of 200 for Case 3. It is found that hypervolume values converge at 30 generations; thus a population size of 200 and 30 generations are used for all the multi-objective optimization cases considered in this paper. The maximum extrusion volume inequality constraint g is implemented using a penalty function. The penalty function gives a fitness disadvantage to the individuals that violate the constraint.

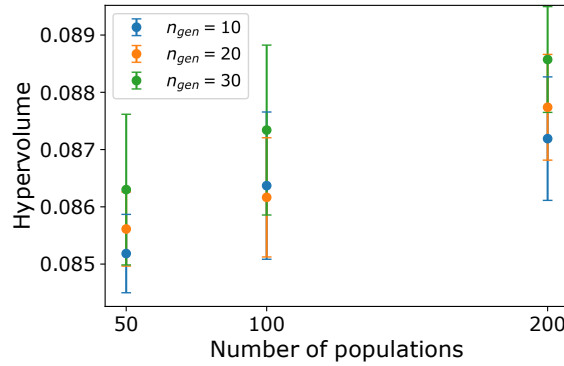


Figure 7.30: Hypervolume values (mean and one standard deviation) of nine different experimental designs of Case 3

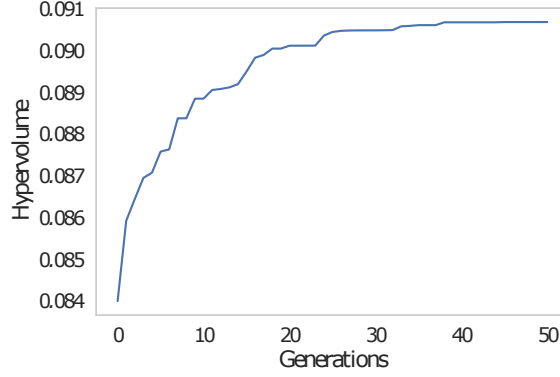


Figure 7.31: Hypervolume values vs. number of generations of Case 3 (Population size 200)

7.3.2.4.1 Case 1

In Case 1, one minus the mean of dimensionless bond quality metric ($1 - \mu_{\hat{b}l}$) and the standard deviation of the dimensionless bond quality metric $\sigma_{\hat{b}l}$ are minimized.

$$\begin{aligned}
 & \underset{\mathbf{x} \in \mathbb{R}^{n_x}}{\text{minimize}} && w_1 (1 - \mu_{\hat{b}l}(\mathbf{x})) + w_2 \sigma_{\hat{b}l}(\mathbf{x}) \\
 & \text{subject to} && g(\mathbf{x}) = 0.8\text{mm} \times V_e \times l_t \leq 24\text{mm}^3/\text{s} \\
 & && 215^\circ\text{C} \leq T_e \leq 280^\circ\text{C} \\
 & && 25\text{mm/s} \leq V_e \leq 45\text{mm/s} \\
 & && l_t \in \{0.42, 0.60, 0.70\}\text{mm}
 \end{aligned} \tag{7.13}$$

The Pareto front obtained for these two objectives is demonstrated in Fig. 7.32 with selected design points A_1 , B_1 , and C_1 for experimental validation. The design points represent three design variables, i.e., nozzle temperature T_e ($^\circ\text{C}$), nozzle speed V_e (mm/s), and layer thickness l_t (mm).

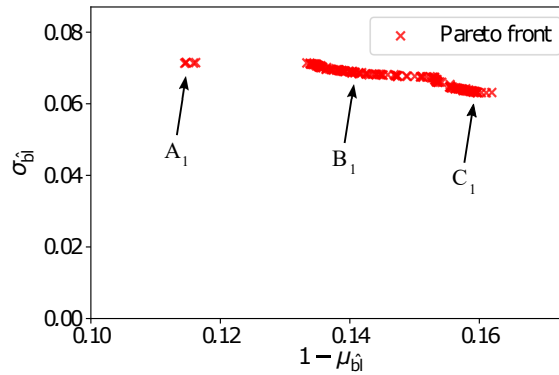


Figure 7.32: Pareto front of Case 1: $A_1 = (T_e$ ($^\circ\text{C}$), V_e (mm/s), l_t (mm)) = (217.09, 26.14, 0.42), $B_1 = (244.54, 29.59, 0.60)$, $C_1 = (219.03, 43.96, 0.42)$

In each generation, several values of the design variables (nozzle temperature, nozzle speed, and layer thickness) are generated and passed to the optimizer to evaluate two objectives. The layer height is restricted to three possible values, 0.42, 0.6, and 0.7 mm, in order to have an integer value for the total number of layers given the desired part thickness of 4.2 mm. A single design can be selected from the optimal designs shown in Fig. 7.32.

7.3.2.4.2 Case 2

The dimensionless mean and standard deviation of part thickness error ($\mu_{\hat{t}_h}$, $\sigma_{\hat{t}_h}$) are minimized in Case 2 using the $\text{BNN}_{\hat{t}_h}$.

$$\begin{aligned}
 & \underset{\mathbf{x} \in \mathbb{R}^{l_x}}{\text{minimize}} && w_1 \mu_{\hat{t}_h}(\mathbf{x}) + w_2 \sigma_{\hat{t}_h}(\mathbf{x}) \\
 & \text{subject to} && g(\mathbf{x}) = 0.8 \text{mm} \times V_e \times l_t \leq 24 \text{mm}^3/\text{s} \\
 & && 215^\circ\text{C} \leq T_e \leq 280^\circ\text{C} \\
 & && 25 \text{mm/s} \leq V_e \leq 45 \text{mm/s} \\
 & && l_t \in \{0.42, 0.60, 0.70\} \text{mm}
 \end{aligned} \tag{7.14}$$

Similar to Case 1, the Pareto front obtained for these two objectives is shown in Fig. 7.33 with selected design points A_2 , B_2 , and C_2 for experimental validation.

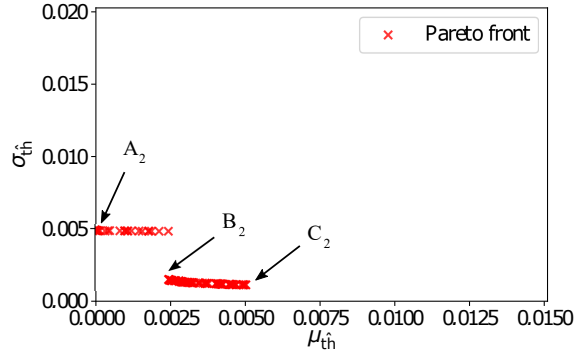


Figure 7.33: Pareto front of Case 2: $A_2 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (272.17, 33.68, 0.42)$, $B_2 = (223.92, 31.02, 0.42)$, $C_2 = (251.41, 36.63, 0.42)$

7.3.2.4.3 Case 3

In this case, one minus the mean of dimensionless bond quality metric ($1 - \mu_{\hat{b}_l}$) and the dimensionless mean of part thickness error $\mu_{\hat{t}_h}$ are minimized by using both models $\text{BNN}_{\hat{b}_l}$ and $\text{BNN}_{\hat{t}_h}$.

$$\begin{aligned}
 & \underset{\mathbf{x} \in \mathbb{R}^{n_x}}{\text{minimize}} && w_1 (1 - \mu_{\hat{b}_l}(\mathbf{x})) + w_2 \mu_{\hat{t}_h}(\mathbf{x}) \\
 & \text{subject to} && g(\mathbf{x}) = 0.8\text{mm} \times V_e \times l_t \leq 24\text{mm}^3/\text{s} \\
 & && 215^\circ\text{C} \leq T_e \leq 280^\circ\text{C} \\
 & && 25\text{mm/s} \leq V_e \leq 45\text{mm/s} \\
 & && l_t \in \{0.42, 0.60, 0.70\}\text{mm}
 \end{aligned} \tag{7.15}$$

The Pareto front of these two objectives is shown in Fig. 7.34 with selected design points A_3 , B_3 , and C_3 for experimental validation.

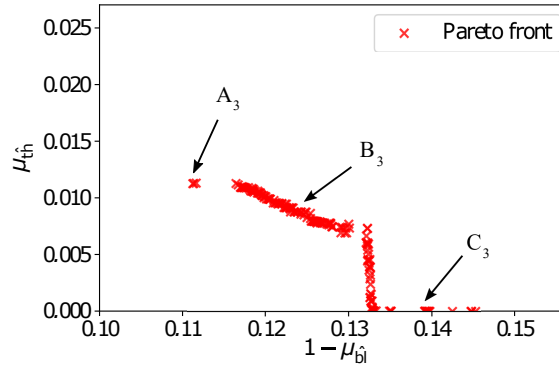


Figure 7.34: Pareto front of Case 3: $A_3 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (217.02, 26.01, 0.42)$, $B_3 = (217.05, 27.84, 0.42)$, $C_3 = (274.04, 34.29, 0.42)$

7.3.2.4.4 Case 4

In this case, one minus the mean of the dimensionless bond quality metric ($\text{Obj}_1 = 1 - \mu_{\hat{b}_l}$), the standard deviation of the dimensionless bond quality metric ($\text{Obj}_2 = \sigma_{\hat{b}_l}$), and the dimensionless mean and standard deviation of part thickness error ($\text{Obj}_3 = \mu_{\hat{t}_h}$ & $\text{Obj}_4 = \sigma_{\hat{t}_h}$) are minimized by using both models $\text{BNN}_{\hat{b}_l}$ and $\text{BNN}_{\hat{t}_h}$. In each generation, a new value of the design variables (T_e ,

V_e, l_t) is generated and passed to the optimizer to evaluate four objectives.

$$\begin{aligned}
 & \underset{\mathbf{x} \in \mathbb{R}^{n_x}}{\text{minimize}} && w_1 (1 - \mu_{\hat{b}_l}(\mathbf{x})) + w_2 \sigma_{\hat{b}_l}(\mathbf{x}) + w_3 \mu_{\hat{t}_h}(\mathbf{x}) + w_4 \sigma_{\hat{t}_h}(\mathbf{x}) \\
 & \text{subject to} && g(\mathbf{x}) = 0.8 \text{mm} \times V_e \times l_t \leq 24 \text{mm}^3/\text{s} \\
 & && 215^\circ\text{C} \leq T_e \leq 280^\circ\text{C} \\
 & && 25 \text{mm/s} \leq V_e \leq 45 \text{mm/s} \\
 & && l_t \in \{0.42, 0.60, 0.70\} \text{mm}
 \end{aligned} \tag{7.16}$$

A plot of the Pareto front is demonstrated in Fig. 7.35, where Obj_1 is displayed on the x-axis, Obj_2 on the y-axis, Obj_3 by the color of the markers, and Obj_4 by the size of the markers. A single design can be selected from the optimal designs shown in Fig. 7.35. The designer can choose a design based on the relative importance of each objective over the others, since slight improvement in one of the objectives may lead to significant degradation in other objectives. From the optimal solutions, three points A_4 , B_4 , and C_4 are chosen for experimental validation.

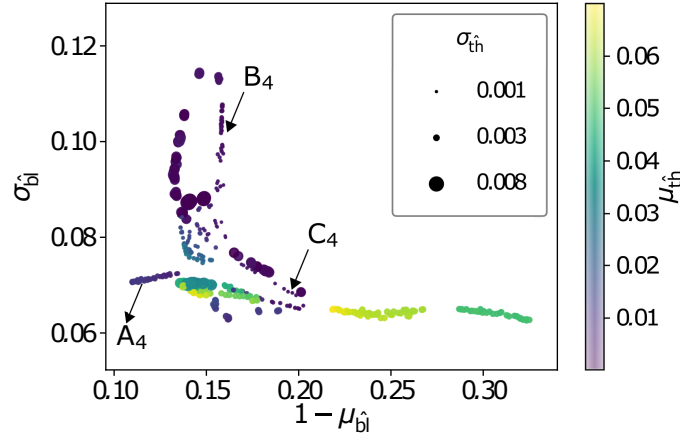


Figure 7.35: Pareto designs for Case 4: $A_4 = (T_e \text{ (}^\circ\text{C)}, V_e \text{ (mm/s)}, l_t \text{ (mm)}) = (217.08, 26.88, 0.42)$, $B_4 = (277.99, 39.41, 0.42)$, $C_4 = (225.34, 31.84, 0.42)$

The optimization results for Case 4 can be illustrated using the parallel coordinate plots shown in Figs. 7.36 and 7.37. The resulting parallel coordinates reflect the Pareto dominance relation between different solutions. In Fig. 7.36, the color map reflects the values of the first objective ($1 - \mu_{\hat{b}_l}$). The red solid line represents the solution that results in the worst mean bond quality within the Pareto front. Note that due to the negative correlation between the first and second objectives (heavily conflicting objectives), worst mean bond quality solutions (i.e., red solid lines) couple with the best (lower) standard deviation bond quality. The mean bond quality starts to improve as the color of the

solid line changes from red to green. In the same manner as Fig. 7.36, Fig. 7.37 also illustrates the optimization results with a color map of the third objective. The red solid lines denote the solutions that yield the poorest mean geometric accuracy within the Pareto front approximations. The mean and standard deviation of the geometric accuracy are partially negatively correlated since there are lines that are parallel between the two axes in parallel coordinates. More specifically, the parallel green solid lines at the bottom of the third and fourth objectives that do not intersect with other lines depict the solutions with moderately good mean and standard deviation of geometric accuracy.

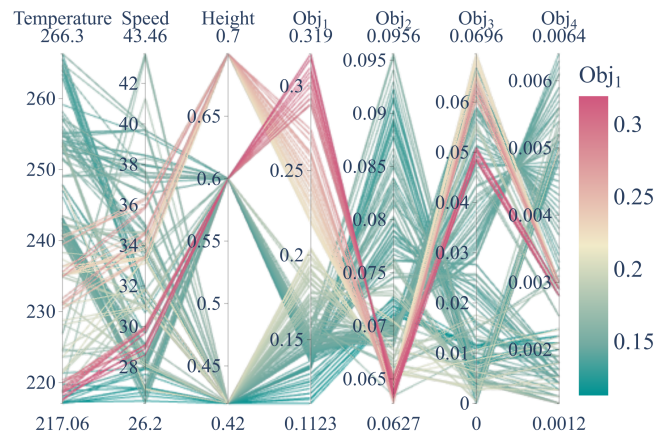


Figure 7.36: Parallel coordinate representation of the model dependencies with a color mapping along the first objective ($1 - \mu_{\hat{1}}$).

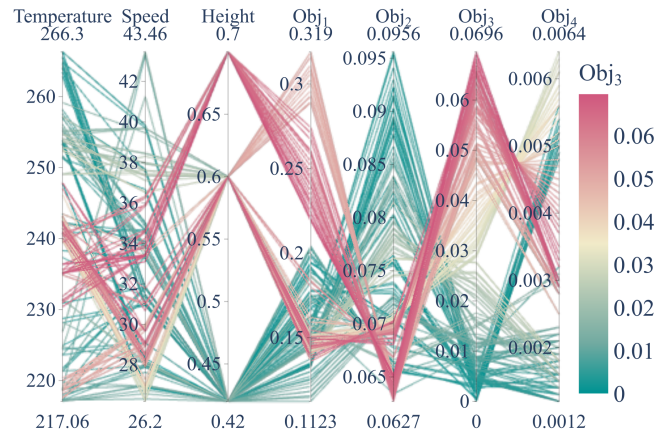


Figure 7.37: Parallel coordinate representation of the model dependencies with a color mapping along the third objective ($\mu_{\hat{3}}$).

The Pearson correlation coefficient between each design variable and/or objective functions is

shown in Fig. 7.38. A correlation coefficient of ± 1 represents a perfect linear relationship. The nozzle temperature (DV_1) is negatively correlated with $Obj_1 = 1 - \mu_{\hat{b}_l}$ and $Obj_3 = \mu_{\hat{h}}$, which means that the mean geometric accuracy and the overall mean bond quality of a part tend to improve with increasing nozzle temperature. Whereas, parts with a greater layer height (DV_3) appear to have a degraded mean geometrical accuracy and mean bond quality. The nozzle speed (DV_2) has a smaller effect on the objective functions. The correlation coefficient between Obj_1 and Obj_3 is calculated as 0.46 indicating that these two objectives do not have a significant relationship, which is also illustrated in Fig. 7.41. Furthermore, the parallel plots and the correlation matrix demonstrate the difficulty in choosing the optimal process parameters. Therefore, the Pareto front is beneficial in finding a design that offers a good trade-off between the objectives.

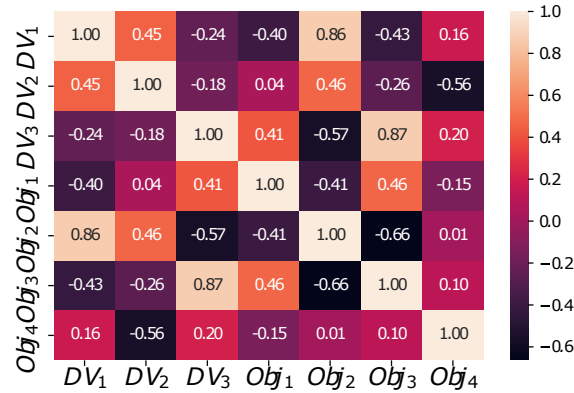


Figure 7.38: The correlation coefficient between design variables and objective functions

The computation time for one optimization is on average ten minutes using a desktop computer (Intel® Xeon® CPU E5-1660 v4@3.20GHz with 32 GB RAM and GPU NVIDIA Quadro K620 with 2 GB).

7.3.2.5 Monte Carlo Optimization

The Pareto front and direct correlations between objective functions obtained from the NSGA-II algorithm in Section 7.3.2.4 are compared with a Monte Carlo sampling (MCS) approach. A relatively large number of Monte Carlo samples (10,000 samples) of design variables from the design space are generated and the objective functions for different cases are predicted using the trained BNN models. Due to computational complexity and difficulty in visualizing the four-dimensional results of Case 4, MCS approach is only employed for bi-objective cases i.e., Cases 1, 2 and 3 (Figs. 7.39, 7.40, 7.41).

The relationship between the two objectives in Case 2 and 3 as shown in Figs. 7.40 and 7.41

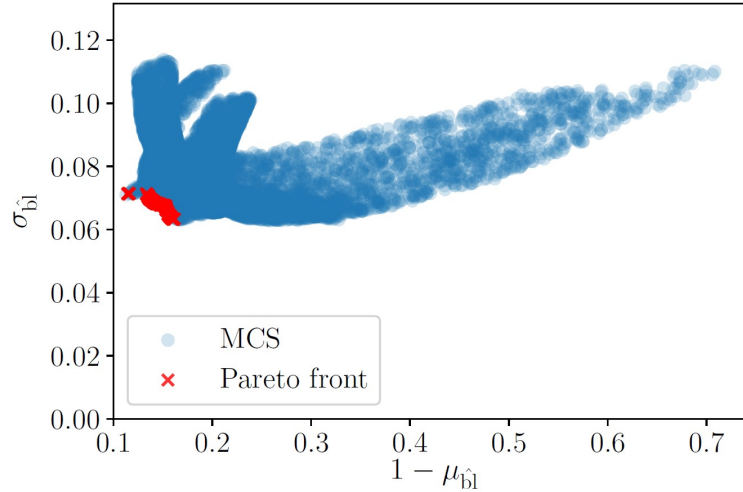


Figure 7.39: Pareto front and MCS results of Case 1

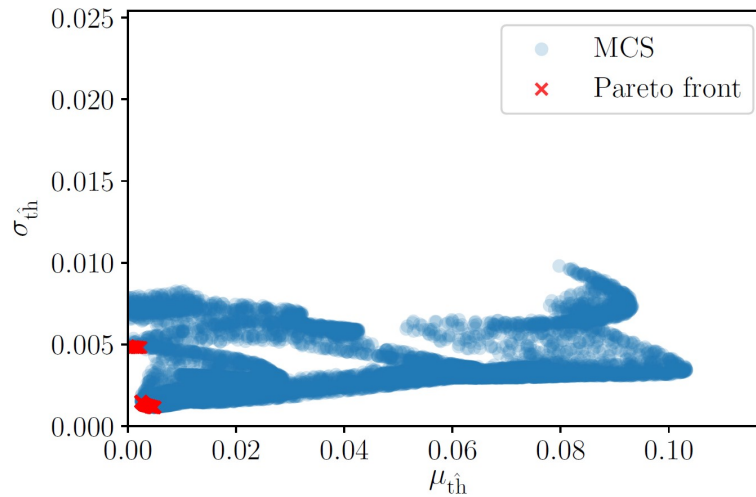


Figure 7.40: Pareto front and MCS results of Case 2

is highly non-linear. The regions with sharp edges generally represent constraint boundaries. The relationship between two physical quantities, i.e., bond quality and geometry accuracy of a part, shown in Fig. 7.41 demonstrates the importance of choosing the optimal process parameters. For some combinations of design variables, both physical quantities may have significantly degraded values, which can be avoided by using the proposed methodology. The Pareto front results shown in Figs. 7.32, 7.33 and 7.34 are superimposed on the MCS results and labeled as red crosses in Figs. 7.39, 7.40, and 7.41. The superimposed Pareto fronts show that the optimization results are in agreement with the MCS results.

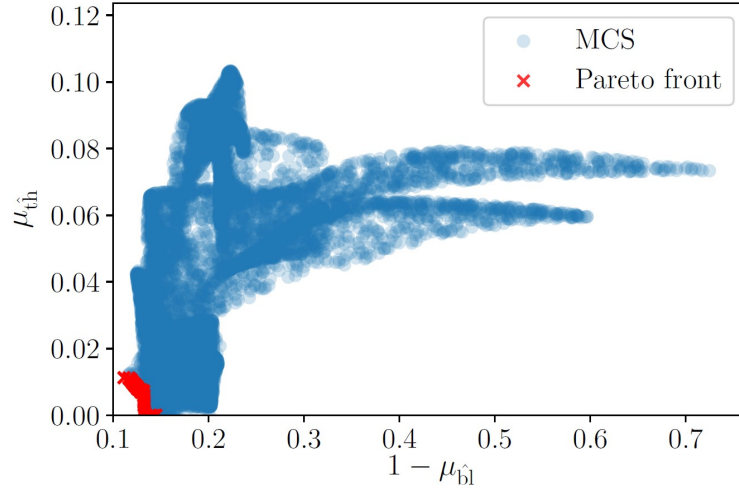


Figure 7.41: Pareto front and MCS results of Case 3

7.3.2.6 Experimental Validation

Three points are chosen for validation from the Pareto fronts for each of the four multi-objective optimization cases considered above. The predicted values of objectives for each point are validated using classical hypothesis testing. The sample size of experiments for each point is 5. The sample size of model predictions for each point is 100, which is produced using MC dropout. For validation of the prediction models, the t -test, which is based on Student's t -distribution, is used to test the hypothesis that the mean of the observations is equal to the mean of the model prediction. Since the calculated p -values for all selected design points ($A_1, B_1, C_1, A_2, B_2, C_2, A_3, B_3, C_3, A_4, B_4, C_4$) are above the significance level $\alpha = 0.05$, the predictions at the optimum solutions agree with the experimental observations. This is the validation of the BNN prediction models, which are trained with the previous experimental data. Note that the training data and validation data are separate.

Next we also validate the optimization result, by demonstrating that the optimum solution gives better part quality than randomly selected process parameter settings. Parts are printed and measured at eleven other process parameter settings selected at random and compared to the part printed with the optimal process parameters of Case 3. The dimensionless mean thickness error ($\mu_{\hat{t}_h}$) and the dimensionless bond quality metric ($1 - \mu_{\hat{b}_l}$) obtained using the randomly selected process parameters are found to be larger than that of the optimal solution (Fig. 7.42), thus clearly demonstrating the effectiveness of the proposed methodology.

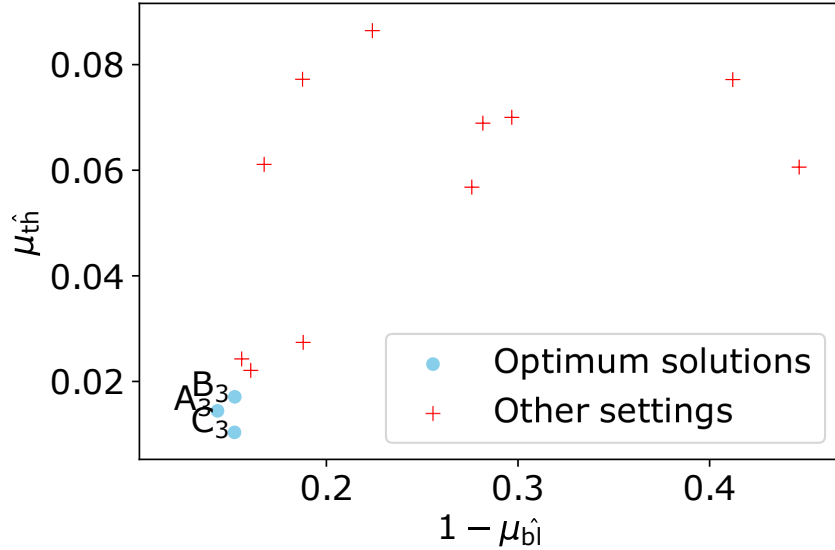


Figure 7.42: Experimental validation of optimal process design (Case 3)

7.3.2.7 Summary

We find that, using the proposed approach, we were able to manufacture FFF parts with better part quality using the optimization results from the Pareto front obtained through the probabilistic data-driven methodology for process parameter optimization. In this section, we obtain probabilistic estimates of both quantities (bond quality and geometrical accuracy) and include the probabilistic estimates within the process design for optimization under uncertainty. The input variability in the experimental data and the model uncertainty are captured through the probabilistic deep learning approach, BNN.

The probabilistic estimates facilitate different options of optimization under uncertainty: optimize only the mean value of an individual quality metric, optimize both the mean and variance of an individual quality metric, optimize the mean values of multiple quality metrics simultaneously, and optimize both the mean values and variances of multiple quality metrics simultaneously. Except the first option, all other options require multi-objective formulations. It is observed that the mean values of both quality metrics (bond length and part thickness accuracy) are optimized with the smallest layer thickness (i.e., 0.42 mm). This is generally expected in AM processes. However, it is not that straightforward to see a similar trend for other parameters. In general, the optimal solutions for the mean of both quality metrics are achieved with larger nozzle temperature and smaller nozzle speed values. However, there are some cases where some other combinations of nozzle temperature and speed (e.g., small temperature and speed) also result in optimal solutions for these two quan-

tities. Thus, the Pareto front is helpful in finding a design that offers a good trade-off between the objectives.

The prediction models are verified using the cross-validation approach as explained in Section 7.3.2.3, and validated against experimental data. The optimization results are validated by printing and measuring the parts corresponding to the points A, B, and C of the Pareto fronts. The results show that the optimum solution using the proposed process optimization framework gives better part quality than randomly selected process parameter settings.

CHAPTER 8

Conclusion

The overall goal of this dissertation is to develop efficient ML models for uncertainty quantification and decision-making under uncertainty with limited available experimental data. In this research, both time-independent and time-dependent systems are considered. Towards this end, this research has investigated physics-informed machine learning (PIML) strategies and their combinations for global sensitivity analysis (GSA), for accurate system response prediction and optimization under uncertainty using both physics knowledge and experimental data. The proposed adaptive sampling and multi-level Bayesian calibration strategies particularly help to approach challenging multivariate time-dependent multi-component systems. A summary of the accomplishments and contributions of this dissertation are outlined below.

8.1 Summary of Contributions

The contributions and accomplishments of this dissertation are outlined as follows.

1. Physics-informed and hybrid machine learning models are developed to allow accurate model predictions even with smaller amounts of experimental data. Three types of strategies are explored to incorporate physics constraints and multi-physics fused filament fabrication (FFF) simulation results into a deep neural network (DNN), thus ensuring consistency with physical laws: (1) incorporate physics constraints within the loss function of the DNN, (2) use physics model outputs as additional inputs to the DNN model, and (3) pre-train a DNN model with physics model input-output and then update it with experimental data. These strategies help to enforce a physically consistent relationship. Thus, the proposed approach helps to fill the physics knowledge gap in the ML model while leveraging the capability of ML to extract complex process-material-geometry relationships, and correcting for the approximation in the physics-based model. (Chapter 3)
2. A robust information fusion and machine learning methodology is developed for sensitivity analysis using both physics knowledge and experimental data, while accounting for model uncertainty. Physics-informed machine learning strategies are investigated to effectively combine the physics-based and experimental information in order to maximize the accuracy of the sensitivity estimate. (Chapter 4)

3. We look to adaptively improve the model by developing an adaptive sampling strategy for training points selection that aims at optimizing resource allocation. An adaptive surrogate modeling method is developed for high-dimensional problems to predict spatio-temporal output quantities of interest (QoIs) using a two-step dimension reduction approach and to improve the surrogate model accuracy with the fewest possible runs of the expensive physics-based model. (Chapter 5)
4. A multi-level Bayesian calibration approach is developed for information fusion from heterogeneous sources to estimate the model parameters with real-time monitoring data of time-dependent multi-component systems, and prediction of the real-time QoIs using the updated model. Multi-level Bayesian calibration is performed to estimate component-level and system-level parameters using experimental data that are obtained at different time instances for different system components. (Chapter 6)
5. Several multi-objective problem formulations are developed for process parameter optimization under uncertainty using both physics-based and data-driven models. The multi-objective process optimization methodology is developed for additive manufacturing (AM) using both physics models and data-driven models. With a focus on FFF AM process, the proposed methodology optimizes the process parameters with the objectives of minimizing the geometric inaccuracy and maximizing the filament bond quality of the manufactured part. Uncertainty in the prediction model is considered in the optimization. Using the predictions from these models, different robust design optimization formulations are investigated. The effectiveness of the proposed methodology is validated by manufacturing the parts at optimal settings and demonstrating the quality improvement. (Chapter 7)

8.2 Future Work

Future research may be pursued in the following directions.

First of all, the data produced by experiments and physics models have different levels of credibility; thus the weighting of the two sources of data needs to be investigated in the future. Also, in order to further evaluate the performance of the proposed PIML strategies and how different combinations of methods affect the accuracy and computational effort, a more complex problem involving several random variables can be considered. Future work can also explore the generalization capabilities of the proposed strategies to parts of different geometry, as well as transfer learning to parts manufactured with different 3D printers and materials.

The proposed PIML-based sensitivity analysis approaches need to be tested for problems with a larger number of dimensions both in the input and output, with multiple combinations to further analyze the convergence of Sobol' index estimates. In order to have a more complete comparison between the GP and DNN models, the proposed approach for GP models can be extended by using different kernels with varying smoothness.

The proposed adaptive sampling technique used randomly generated points to sequentially select the next training points. Future work can explore an implementation of a rigorous optimization strategy for the same purpose. The reliability of the adaptive sampling techniques in terms of reproducibility of results also needs to be considered. Future work can explore other large-scale engineering examples in order to further evaluate the performance of the proposed method, gain more insight about the method's behavior for different problems, and further improve the methodology.

The proposed multi-level online Bayesian calibration approach could support building a Digital Twin that contains all the information about the engineering system and is continuously updated with real-time information. The Digital Twin can be used to (1) assess the current condition and capabilities of the system, (2) predict the future condition and capabilities of the system, and (3) provide information for decision-making related to system health management (inspection, maintenance, and repair). Future work should conduct more experiments at several spatio-temporal locations to obtain sharper posterior estimates by fusing more information from multiple levels. Further investigation is needed regarding resource allocation for data collection in the context of multi-level calibration and its connection to the preferred calibration strategy. In the future, a more sophisticated and numerically robust method can be used for joint state-parameter estimation, in the presence of time-dependent system states.

In this dissertation, multi-objective decision-making under uncertainty was considered. The proposed approach helps to replace the trial-and-error experimental approach with a model-based process parameter optimization under uncertainty. Future work can also extend this framework to process control under uncertainty, thus further reducing the variability in the QoIs in order to achieve various product quality objectives.

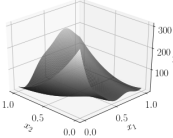
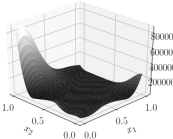
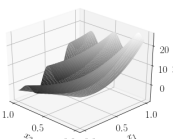
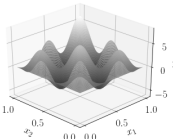
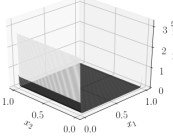
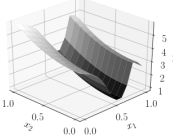
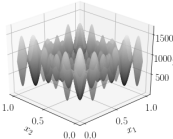
Appendix A

Appendix

A.1 Benchmark Test Functions

Two-dimensional benchmark functions used in this paper to evaluate the performance of the proposed method and compare its performance to existing adaptive surrogate model improvement methods are defined in Table A.1.

Table A.1: Test problems

<p>Branin function</p> $y = \left[x_2 - 5 \left(\frac{x_1}{2\pi} \right)^2 + \frac{5x_1}{\pi} - 6 \right] + 10 \left(1 - \frac{1}{8\pi} \right) \cos(x_1) + 10$ <p>$x_1 \in [-5, 10], x_2 \in [0, 15]$</p>	
<p>Goldstein & Price function</p> $y = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \right] \times \left[30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \right]$ <p>$x_1, x_2 \in [-2, 2]$</p>	
<p>Sasena's function</p> $y = 2 + 0.01(x_2 - x_1^2)^2 + (1 - x_1)^2 + 2(2 - x_2) + 7\sin(0.5x_1) \sin(0.7x_1x_2)$ <p>$x_1, x_2 \in [0, 5]$</p>	
<p>Alpine function</p> $y = \sin x_1 \sin x_2 \sqrt{ x_1 x_2 }$ <p>$x_1, x_2 \in [0, 10]$</p>	
<p>Modified form of function in Meckesheimer et al. [189]</p> $y = e^{(x_1 - x_2)^2} + e^{(10 - x_1)^2} - x_1 x_2$ <p>$x_1, x_2 \in [0, 10]$</p>	
<p>Modified form of function in Jin et al. [127]</p> $y = \cos(10x_1^2) + 3.1 x_1 - 0.7 + 2x_1^2 + \sin\left(\frac{1}{ x_1 - 0.7 + 0.31}\right) + 2x_2^2$ <p>$x_1, x_2 \in [0, 1]$</p>	
<p>Schwefel function</p> $y = 418.9829n - \sum_{i=1}^n (x_i \sin \sqrt{ x_i })$ <p>$x_i \in [-500, 500], i = 1, \dots, n, n = 2$</p>	

REFERENCES

- [1] V.K. Hombal and S. Mahadevan. Surrogate modeling of 3d crack growth. *International Journal of Fatigue*, 47:90–99, 2013. ISSN 0142-1123. doi: <https://doi.org/10.1016/j.ijfatigue.2012.07.012>. URL <https://www.sciencedirect.com/science/article/pii/S0142112312002393>.
- [2] Paromita Nath, Zhen Hu, and Sankaran Mahadevan. Sensor placement for calibration of spatially varying model parameters. *Journal of Computational Physics*, 343:150–169, 2017.
- [3] Bruce M Davis and Kent A Greenes. Estimation using spatially distributed multivariate data: an example with coal quality. *Journal of the International Association for Mathematical Geology*, 15(2):287–300, 1983.
- [4] Leon E Borgman and RB Frahme. A case study: multivariate properties of bentonite in northeastern wyoming. In *Advanced Geostatistics in the mining industry*, pages 381–390. Springer, 1976.
- [5] Yulin Guo, Sankaran Mahadevan, Shunsaku Matsumoto, Shunsuke Taba, and Daigo Watanabe. Surrogate modeling with high-dimensional input and output. In *AIAA Scitech 2021 Forum*, page 0182, 2021.
- [6] Zhen Hu and Sankaran Mahadevan. A surrogate modeling approach for reliability analysis of a multidisciplinary system with spatio-temporal output. *Structural and Multidisciplinary Optimization*, 56(3):553–569, 2017.
- [7] G Berkooz, P Holmes, and J L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, 1993. doi: 10.1146/annurev.fl.25.010193.002543.
- [8] Youssef M Marzouk and Habib N Najm. Dimensionality reduction and polynomial chaos acceleration of bayesian inference in inverse problems. *Journal of Computational Physics*, 228(6):1862–1902, 2009. doi: 10.1016/j.jcp.2008.11.024.
- [9] Paul G. Constantine, Michael S. Eldred, and Eric T. Phipps. Sparse pseudospectral approximation method. *Computer Methods in Applied Mechanics and Engineering*, 229-232:1–12, 2012. ISSN 0045-7825. doi: <https://doi.org/10.1016/j.cma.2012.03.019>.
- [10] Patrick R. Conrad and Youssef M. Marzouk. Adaptive smolyak pseudospectral approximations. *SIAM Journal on Scientific Computing*, 35(6):A2643–A2670, 2013. doi: 10.1137/120890715.
- [11] Xiang Ma and Nicholas Zabaras. An efficient bayesian inference approach to inverse problems based on an adaptive sparse grid collocation method. *Inverse Problems*, 25(3):035013, 2009. doi: 10.1088/0266-5611/25/3/035013.
- [12] Paul G Constantine. *Active subspaces: Emerging ideas for dimension reduction in parameter studies*. SIAM, 2015.
- [13] Olivier Zahm, Paul G. Constantine, Clémentine Prieur, and Youssef M. Marzouk. Gradient-based dimension reduction of multivariate vector-valued functions. *SIAM Journal on Scientific Computing*, 42(1):A534–A558, 2020. doi: 10.1137/18M1221837.
- [14] Manav Vohra, Paromita Nath, Sankaran Mahadevan, and Yung-Tsun Tina Lee. Fast surrogate modeling using dimensionality reduction in model inputs and field output: Application to additive manufacturing. *Reliability Engineering & System Safety*, 201:106986, 2020. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.ress.2020.106986>.

- [15] Andrew White, Sankaran Mahadevan, Zachary Grey, Jason Schmucker, and Alexander Karl. Efficient calibration of a turbine disc heat transfer model under uncertainty. *Journal of Thermophysics and Heat Transfer*, 35(2):234–244, 2021. doi: 10.2514/1.T6047.
- [16] Jerome Sacks, William J Welch, Toby J Mitchell, and Henry P Wynn. Design and analysis of computer experiments. *Statistical science*, pages 409–423, 1989.
- [17] Michael D McKay, Richard J Beckman, and William J Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979. doi: 10.2307/1268522.
- [18] Saideep Nannapaneni and Sankaran Mahadevan. Manufacturing process evaluation under uncertainty: A hierarchical bayesian network approach. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 50084, page V01BT02A026. American Society of Mechanical Engineers, 2016. doi: 10.1115/DETC2016-59226.
- [19] Erin C DeCarlo, Benjamin P Smarslok, and Sankaran Mahadevan. Segmented bayesian calibration of multidisciplinary models. *AIAA Journal*, 54(12):3727–3741, 2016. doi: 10.2514/1.J054960.
- [20] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global sensitivity analysis: the primer*. John Wiley & Sons, 2008.
- [21] A. Saltelli, S. Tarantola, and K. P.-S. Chan. A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics*, 41(1):39–56, 1999. doi: 10.1080/00401706.1999.10485594.
- [22] Thierry A. Mara and Stefano Tarantola. Variance-based sensitivity indices for models with dependent inputs. *Reliability Engineering & System Safety*, 107:115 – 121, 2012. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.res.2011.08.008>. SAMO 2010.
- [23] Emanuele Borgonovo and Elmar Plischke. Sensitivity analysis: A review of recent advances. *European Journal of Operational Research*, 248(3):869 – 887, 2016. ISSN 0377-2217. doi: <https://doi.org/10.1016/j.ejor.2015.06.032>.
- [24] Scott Ferson and W Troy Tucker. Sensitivity analysis using probability bounding. *Reliability Engineering & System Safety*, 91(10-11):1435–1442, 2006. doi: 10.1016/j.res.2005.11.052.
- [25] Michael Oberguggenberger, Julian King, and Bernhard Schmelzer. Classical and imprecise probability methods for sensitivity analysis in engineering: A case study. *International Journal of Approximate Reasoning*, 50(4):680–693, 2009. doi: 10.1016/j.ijar.2008.09.004.
- [26] Berkcan Kapusuzoglu, Matthew Sato, Sankaran Mahadevan, and Paul Witherell. Process Optimization under Uncertainty for Improving the Bond Quality of Polymer Filaments in Fused Filament Fabrication. *Journal of Manufacturing Science and Engineering*, pages 1–46, 08 2020. ISSN 1087-1357. URL <https://doi.org/10.1115/1.4048073>.
- [27] Paromita Nath, Joseph D. Olson, Sankaran Mahadevan, and Yung-Tsun Tina Lee. Optimization of fused filament fabrication process parameters under uncertainty to maximize part geometry accuracy. *Additive Manufacturing*, 35:101331, 2020. ISSN 2214-8604. doi: 10.1016/j.addma.2020.101331.
- [28] Donghong Ding, Zengxi Pan, Dominic Cuiuri, Huijun Li, Stephen van Duin, and Nathan Larkin. Bead modelling and implementation of adaptive mat path in wire and arc additive manufacturing. *Robotics and Computer-Integrated Manufacturing*, 39:32–42, 2016. doi: 10.1016/j.rcim.2015.12.004.

- [29] Mehrshad Mehrpouya, Annamaria Gisario, Atabak Rahimzadeh, Mohammadreza Nematollahi, Keyvan Safaei Baghbaderani, and Mohammad Elahinia. A prediction model for finding the optimal laser parameters in additive manufacturing of niti shape memory alloy. *The International Journal of Advanced Manufacturing Technology*, 105(11):4691–4699, 2019. doi: 10.1007/s00170-019-04596-z.
- [30] Dongsen Ye, Jerry Ying Hsi Fuh, Yingjie Zhang, Geok Soon Hong, and Kungpeng Zhu. In situ monitoring of selective laser melting using plume and spatter signatures by deep belief networks. *ISA transactions*, 81:96–104, 2018. doi: 10.1016/j.isatra.2018.07.021.
- [31] Aniruddha Gaikwad, Farhad Imani, Hui Yang, Edward Reutzel, and Prahalada Rao. In situ monitoring of thin-wall build quality in laser powder bed fusion using deep learning. *Smart and Sustainable Manufacturing Systems*, 3, 2019. doi: 10.1520/SSMS20190027.
- [32] Paromita Nath, Zhen Hu, and Sankaran Mahadevan. Uncertainty quantification of grain morphology in laser direct metal deposition. *Modelling and Simulation in Materials Science and Engineering*, 27(4):044003, apr 2019. doi: 10.1088/1361-651x/ab1676.
- [33] Mohamad Mahmoudi, Gustavo Tapia, Kubra Karayagiz, Brian Franco, Ji Ma, Raymundo Arroyave, Ibrahim Karaman, and Alaa Elwany. Multivariate calibration and experimental validation of a 3D finite element thermal model for laser powder bed fusion metal additive manufacturing. *Integrating Materials and Manufacturing Innovation*, 7(3):116–135, 2018. doi: 10.1007/s40192-018-0113-z.
- [34] Anindya Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, pages 808–817, 2000.
- [35] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011. doi: 10.1137/090771806.
- [36] Carl Edward Rasmussen. *Gaussian Processes in Machine Learning*, pages 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-28650-9. doi: 10.1007/978-3-540-28650-9_4.
- [37] Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, Cambridge, 2006.
- [38] Berkcan Kapusuzoglu and Sankaran Mahadevan. Information fusion and machine learning for sensitivity analysis using physics knowledge and experimental data. *Reliability Engineering & System Safety*, 214:107712, 2021. ISSN 0951-8320. doi: 10.1016/j.ress.2021.107712.
- [39] Vadiraj Hombal and Sankaran Mahadevan. Bias minimization in gaussian process surrogate modeling for uncertainty quantification. *Visualization of Mechanical Processes: An International Online Journal*, 1(4), 2011. doi: 10.1615/Int.J.UncertaintyQuantification.2011003343.
- [40] John S Denker and Yann LeCun. Transforming neural-net output levels to probability distributions. In *Advances in neural information processing systems*, pages 853–859, 1991.
- [41] David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.
- [42] R. M. Neal. *Bayesian learning for neural networks*. Lecture Notes in Statistics, 1996.
- [43] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1613–1622. PMLR, 2015.
- [44] Mark A Beaumont, Wenyang Zhang, and David J Balding. Approximate bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002. doi: 10.1371/journal.pcbi.1002803.

- [45] Walter R Gilks. Markov chain monte carlo. *Encyclopedia of Biostatistics*, 4, 2005. doi: 10.1002/0470011815.b2a14021.
- [46] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002. doi: 10.1109/78.978374.
- [47] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.
- [48] Eric VanDerHorn and Sankaran Mahadevan. Bayesian model updating with summarized statistical and reliability data. *Reliability Engineering & System Safety*, 172:12–24, 2018. doi: 10.1016/j.ress.2017.11.023.
- [49] Pranav M Karve, Yulin Guo, Berkcan Kapusuzoglu, Sankaran Mahadevan, and Mulugeta A Haile. Digital twin approach for damage-tolerant mission planning under uncertainty. *Engineering Fracture Mechanics*, 225:106766, 2020. doi: 10.1016/j.engfracmech.2019.106766.
- [50] 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. doi: 10.1111/1467-9868.00294.
- [51] You Ling, Joshua Mullins, and Sankaran Mahadevan. Selection of model discrepancy priors in Bayesian calibration. *Journal of Computational Physics*, 276:665 – 680, 2014. ISSN 0021-9991. doi: <https://doi.org/10.1016/j.jcp.2014.08.005>.
- [52] Abhinav Subramanian and Sankaran Mahadevan. Bayesian estimation of discrepancy in dynamics model prediction. *Mechanical Systems and Signal Processing*, 123:351–368, 2019. ISSN 0888-3270. doi: <https://doi.org/10.1016/j.ymsp.2019.01.014>.
- [53] Abhinav Subramanian and Sankaran Mahadevan. Model error propagation from experimental to prediction configuration. *Journal of Computational Physics*, 443:110529, 2021. ISSN 0021-9991. doi: <https://doi.org/10.1016/j.jcp.2021.110529>.
- [54] Rudraprasad Bhattacharyya, Sankaran Mahadevan, and Prodyot K Basu. Computationally efficient multiscale modeling for probabilistic analysis of cfrp composites with micro-scale spatial randomness. *Composite Structures*, 280:114884, 2022. doi: 10.1016/j.compstruct.2021.114884.
- [55] W K Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- [56] George Casella and Edward I George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.
- [57] Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–767, June 2003. ISSN 0090-5364. doi: 10.1214/aos/1056562461.
- [58] Erich Novak and Henryk Woźniakowski. Approximation of infinitely differentiable multivariate functions is intractable. *Journal of Complexity*, 25(4):398–404, 2009. doi: 10.1016/j.jco.2008.11.002.
- [59] Jun S Liu. Metropolisized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and computing*, 6(2):113–119, 1996. doi: 10.1007/BF00162521.
- [60] Nikolas Kantas, Arnaud Doucet, Sumeetpal S Singh, Jan Maciejowski, and Nicolas Chopin. On particle methods for parameter estimation in state-space models. *Statistical science*, 30(3):328–351, 2015. doi: 10.1214/14-STS511.

- [61] Hamid Moradkhani, Kuo-Lin Hsu, Hoshin Gupta, and Soroosh Sorooshian. Uncertainty assessment of hydrologic model states and parameters: Sequential data assimilation using the particle filter. *Water Resources Research*, 41(5), 2005. doi: <https://doi.org/10.1029/2004WR003604>.
- [62] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988.
- [63] Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and Computers in Simulation*, 55(1-3):271–280, 2001.
- [64] Bruno Sudret. Global sensitivity analysis using polynomial chaos expansions. *Reliability Engineering & System Safety*, 93(7):964 – 979, 2008. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.res.2007.04.002>. Bayesian Networks in Dependability.
- [65] Wei Chen, Ruichen Jin, and Agus Sudjianto. Analytical Variance-Based Global Sensitivity Analysis in Simulation-Based Design Under Uncertainty. *Journal of Mechanical Design*, 127(5):875–886, 12 2004. ISSN 1050-0472. doi: 10.1115/1.1904642.
- [66] S. Tarantola, D. Gatelli, and T.A. Mara. Random balance designs for the estimation of first order global sensitivity indices. *Reliability Engineering & System Safety*, 91(6):717 – 727, 2006. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.res.2005.06.003>.
- [67] F. E. Satterthwaite. Random balance experimentation. *Technometrics*, 1(2):111–137, 1959. doi: 10.1080/00401706.1959.10489853.
- [68] Loïc Le Gratiet, Stefano Marelli, and Bruno Sudret. Metamodel-based sensitivity analysis: Polynomial chaos expansions and gaussian processes. *Handbook of Uncertainty Quantification*, page 1–37, 2015. doi: 10.1007/978-3-319-11259-6_38-1. URL http://dx.doi.org/10.1007/978-3-319-11259-6_38-1.
- [69] Amandine Marrel, Bertrand Iooss, Sébastien Da Veiga, and Mathieu Ribatet. Global sensitivity analysis of stochastic computer models with joint metamodels. *Statistics and Computing*, 22(3):833–847, 2012. ISSN 1573-1375. doi: 10.1007/s11222-011-9274-8. URL <https://doi.org/10.1007/s11222-011-9274-8>.
- [70] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2):619–644, 2002. doi: 10.1137/S1064827501387826.
- [71] Alexandre Janon, Maelle Nodet, and Clementine Prieur. Uncertainties assessment in global sensitivity indices estimation from metamodels. *International Journal for Uncertainty Quantification*, 4(1):21–36, 2014. ISSN 2152-5080.
- [72] Zhen Hu and Xiaoping Du. Mixed Efficient Global Optimization for Time-Dependent Reliability Analysis. *Journal of Mechanical Design*, 137(5), 05 2015. ISSN 1050-0472. doi: 10.1115/1.4029520. URL <https://doi.org/10.1115/1.4029520>. 051401.
- [73] SF Costa, FM Duarte, and JA Covas. Estimation of filament temperature and adhesion development in fused deposition techniques. *Journal of Materials Processing Technology*, 245: 167–179, 2017. doi: 10.1016/j.jmatprotec.2017.02.026.
- [74] Richard S Palais and Robert Andrew Palais. *Differential equations, mechanics, and computation*, volume 51. American Mathematical Soc., 2009.
- [75] Ondřej Pokluda, Céline T Bellehumeur, and John Vlachopoulos. Modification of frenkel’s model for sintering. *AIChE journal*, 43(12):3253–3256, 1997. doi: 10.1002/aic.690431213.
- [76] N. Rosenzweig and M. Narkis. Sintering rheology of amorphous polymers. *Polymer Engineering & Science*, 21(17):1167–1170, 1981. doi: 10.1002/pen.760211709.

- [77] JJ Frenkel. Viscous flow of crystalline bodies under the action of surface tension. *J. phys.*, 9: 385, 1945.
- [78] Céline Bellehumeur, Longmei Li, Qian Sun, and Peihua Gu. Modeling of bond formation between polymer filaments in the fused deposition modeling process. *Journal of Manufacturing Processes*, 6(2):170–178, 2004. doi: 10.1016/S1526-6125(04)70071-7.
- [79] S. Garzon-Hernandez, D. Garcia-Gonzalez, A. Jérusalem, and A. Arias. Design of FDM 3D printed polymers: An experimental-modelling methodology for the prediction of mechanical properties. *Materials & Design*, 188:108414, 2020. ISSN 0264-1275. doi: <https://doi.org/10.1016/j.matdes.2019.108414>.
- [80] Pavan Kumar Gurralla and Srinivasa Prakash Regalla. Part strength evolution with bonding between filaments in fused deposition modelling: This paper studies how coalescence of filaments contributes to the strength of final fdm part. *Virtual and Physical Prototyping*, 9(3): 141–149, 2014. doi: 10.1080/17452759.2014.913400.
- [81] Tarasankar Debroy, Wei Zhang, J Turner, and Sudarsanam Suresh Babu. Building digital twins of 3d printing machines. *Scripta Materialia*, 135:119–124, 2017. doi: 10.1016/j.scriptamat.2016.12.005.
- [82] R W Hopper. Coalescence of two equal cylinders: exact results for creeping viscous plane flow driven by capillarity. *J. Am. Ceram. Soc.; (United States)*, 67:12, 12 1984. doi: 10.1111/j.1151-2916.1984.tb19692.x.
- [83] Mojtaba Khanzadeh, Sudipta Chowdhury, Mohammad Marufuzzaman, Mark A Tschopp, and Linkan Bian. Porosity prediction: Supervised-learning of thermal history for direct laser deposition. *Journal of manufacturing systems*, 47:69–82, 2018. doi: 10.1016/j.jmsy.2018.04.001.
- [84] Ohyung Kwon, Hyung Giun Kim, Wonrae Kim, Gun-Hee Kim, and Kangil Kim. A convolutional neural network for prediction of laser power using melt-pool images in laser powder bed fusion. *IEEE Access*, 2020. doi: 10.1109/ACCESS.2020.2970026.
- [85] Bin Zhang, Shunyu Liu, and Yung C. Shin. In-process monitoring of porosity during laser additive manufacturing process. *Additive Manufacturing*, 28:497 – 505, 2019. ISSN 2214-8604. doi: <https://doi.org/10.1016/j.addma.2019.05.030>.
- [86] Anuj Karpatne, William Watkins, Jordan Read, and Vipin Kumar. Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling. *arXiv e-prints*, art. arXiv:1710.11431, October 2017.
- [87] Xiaowei Jia, Jared Willard, Anuj Karpatne, Jordan S Read, Jacob A Zwart, Michael Steinbach, and Vipin Kumar. Physics-Guided Machine Learning for Scientific Discovery: An Application in Simulating Lake Temperature Profiles. *arXiv e-prints*, art. arXiv:2001.11086, January 2020.
- [88] Caroline A Schneider, Wayne S Rasband, and Kevin W Eliceiri. Nih image to imagej: 25 years of image analysis. *Nature methods*, 9(7):671, 2012.
- [89] Q Sun, GM Rizvi, CT Bellehumeur, and P Gu. Effect of processing conditions on the bonding quality of fdm polymer filaments. *Rapid Prototyping Journal*, 14(2):72–80, 2008. doi: 10.1108/13552540810862028.
- [90] Francois Chollet et al. Keras, 2015. URL <https://github.com/fchollet/keras>.
- [91] Loic Le Gratiet, Claire Cannamela, and Bertrand Iooss. A bayesian approach for global sensitivity analysis of (multifidelity) computer codes. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):336–363, 2014. doi: <https://doi.org/10.1137/130926869>.

- [92] Zhen Hu and Sankaran Mahadevan. Probability models for data-driven global sensitivity analysis. *Reliability Engineering & System Safety*, 187:40 – 57, 2019. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.ress.2018.12.003>. Sensitivity Analysis of Model Output.
- [93] Amandine Marrel, Bertrand Iooss, François Van Dorpe, and Elena Volkova. An efficient methodology for modeling complex computer codes with gaussian processes. *Computational Statistics & Data Analysis*, 52(10):4731 – 4744, 2008. ISSN 0167-9473. doi: <https://doi.org/10.1016/j.csda.2008.03.026>.
- [94] A. O’Hagan. Bayesian analysis of computer code outputs: A tutorial. *Reliability Engineering & System Safety*, 91(10):1290 – 1300, 2006. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.ress.2005.11.025>. The Fourth International Conference on Sensitivity Analysis of Model Output (SAMO 2004).
- [95] S. Sankararaman and S. Mahadevan. Separating the contributions of variability and parameter uncertainty in probability distributions. *Reliability Engineering & System Safety*, 112:187 – 199, 2013. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.ress.2012.11.024>.
- [96] Chenzhao Li and Sankaran Mahadevan. Relative contributions of aleatory and epistemic uncertainty sources in time series prediction. *International Journal of Fatigue*, 82:474 – 486, 2016. ISSN 0142-1123. doi: <https://doi.org/10.1016/j.ijfatigue.2015.09.002>.
- [97] Jeremy E. Oakley and Anthony O’Hagan. Probabilistic sensitivity analysis of complex models: a bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769, 2004. doi: 10.1111/j.1467-9868.2004.05304.x.
- [98] Amandine Marrel, Bertrand Iooss, Béatrice Laurent, and Olivier Roustant. Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety*, 94(3):742 – 751, 2009. ISSN 0951-8320. doi: <https://doi.org/10.1016/j.ress.2008.07.008>.
- [99] Vincent Ginot, Sabrina Gaba, Rémy Beaudouin, Franck Aries, and Hervé Monod. Combined use of local and anova-based global sensitivity analyses for the investigation of a stochastic dynamic model: Application to the case study of an individual-based model of a fish population. *Ecological Modelling*, 193(3):479 – 491, 2006. ISSN 0304-3800. doi: <https://doi.org/10.1016/j.ecolmodel.2005.08.025>. URL <http://www.sciencedirect.com/science/article/pii/S0304380005004084>.
- [100] G. E. B. Archer, A. Saltelli, and I. M. Sobol. Sensitivity measures, anova-like techniques and the use of bootstrap. *Journal of Statistical Computation and Simulation*, 58(2):99–120, 1997. doi: 10.1080/00949659708811825.
- [101] Chenzhao Li and Sankaran Mahadevan. An efficient modularized sample-based method to estimate the first-order sobol’ index. *Reliability Engineering & System Safety*, 153:110–121, 2016. doi: 10.1016/j.ress.2016.04.012.
- [102] Erin C DeCarlo, Sankaran Mahadevan, and Benjamin P Smarslok. Efficient global sensitivity analysis with correlated variables. *Structural and Multidisciplinary Optimization*, 58(6):2325–2340, 2018. doi: 10.1007/s00158-018-2077-1.
- [103] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. Integrating physics-based modeling with machine learning: A survey, 2020.
- [104] Berkcan Kapusuzoglu and Sankaran Mahadevan. Physics-informed and hybrid machine learning in additive manufacturing: Application to fused filament fabrication. *JOM*, 72:4695–4705, 2020. doi: 10.1007/s11837-020-04438-4.
- [105] Areski Cousin, Hassan Maatouk, and Didier Rulliére. Kriging of financial term-structures. *European Journal of Operational Research*, 255(2):631 – 648, 2016. ISSN 0377-2217. doi: <https://doi.org/10.1016/j.ejor.2016.05.057>. URL <http://www.sciencedirect.com/science/article/pii/S0377221716303940>.

- [106] Sébastien Da Veiga and Amandine Marrel. Gaussian process modeling with inequality constraints. *Annales de la Faculté des sciences de Toulouse : Mathématiques*, Ser. 6, 21(3): 529–555, 2012. doi: 10.5802/afst.1344. URL http://www.numdam.org/item/AFST_2012__6_21_3_529_0.
- [107] Shirin Golchi, Derek R Bingham, Hugh Chipman, and David A Campbell. Monotone emulation of computer experiments. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):370–392, 2015. URL <https://doi.org/10.1137/140976741>.
- [108] Andrés F López-Lopera, François Bachoc, Nicolas Durrande, and Olivier Roustant. Finite-dimensional gaussian approximation with linear inequality constraints. *SIAM/ASA Journal on Uncertainty Quantification*, 6(3):1224–1255, 2018. URL <https://doi.org/10.1137/17M1153157>.
- [109] Jaakko Riihimäki and Aki Vehtari. Gaussian processes with monotonicity information. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 645–652, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. JMLR Workshop and Conference Proceedings.
- [110] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *Siam Review*, 60(3):550–591, 2018.
- [111] Ghina N Absi and Sankaran Mahadevan. Multi-fidelity approach to dynamics model calibration. *Mechanical Systems and Signal Processing*, 68:189–206, 2016. doi: <https://doi.org/10.1016/j.ymssp.2015.07.019>.
- [112] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999. doi: <https://doi.org/10.1023/A:1007665907178>.
- [113] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995. doi: 10.1007/BF00994018.
- [114] Thomas Gerstner and Michael Griebel. Numerical integration using sparse grids. *Numerical Algorithms*, 18(3):209–232, 1998.
- [115] Baskar Ganapathysubramanian and Nicholas Zabaras. Sparse grid collocation schemes for stochastic natural convection problems. *Journal of Computational Physics*, 225(1):652–685, 2007.
- [116] Patrick R Conrad and Youssef M Marzouk. Adaptive smolyak pseudospectral approximations. *SIAM Journal on Scientific Computing*, 35(6):A2643–A2670, 2013.
- [117] Géraud Blatman and Bruno Sudret. Adaptive sparse polynomial chaos expansion based on least angle regression. *Journal of Computational Physics*, 230(6):2345–2367, 2011.
- [118] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1):37–52, 1987. ISSN 0169-7439. doi: [https://doi.org/10.1016/0169-7439\(87\)80084-9](https://doi.org/10.1016/0169-7439(87)80084-9). *Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists*.
- [119] Donald E Myers. Matrix formulation of co-kriging. *Journal of the International Association for Mathematical Geology*, 14(3):249–257, 1982.
- [120] Christian Gogu and Jean-Charles Passieux. Efficient surrogate construction by combining response surface methodology and reduced order modeling. *Structural and Multidisciplinary Optimization*, 47(6):821–837, 2013.

- [121] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011. doi: 10.1137/090771806.
- [122] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006. doi: 10.1137/S0097539704442696.
- [123] Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pages 143–152, 2006. doi: 10.1109/FOCS.2006.37.
- [124] George EP Box and J Stuart Hunter. The 2 k—p fractional factorial designs. *Technometrics*, 3(3):311–351, 1961.
- [125] Art B Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, pages 439–452, 1992.
- [126] Mark E Johnson, Leslie M Moore, and Donald Ylvisaker. Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, 26(2):131–148, 1990.
- [127] Ruichen Jin, Wei Chen, and Agus Sudjianto. On sequential sampling for global metamodeling in engineering design. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 36223, pages 539–548, 2002.
- [128] Genzi Li, Vikrant Aute, and Shapour Azarm. An accumulative error based adaptive design of experiments for offline metamodeling. *Structural and Multidisciplinary Optimization*, 40(1-6): 137, 2010.
- [129] Jack PC Kleijnen and Wim CM Van Beers. Application-driven sequential designs for simulation experiments: Kriging metamodelling. *Journal of the Operational Research Society*, 55 (8):876–883, 2004.
- [130] David A Romero, Cristina H Amon, and Susan Finger. On adaptive sampling for single and multi-response bayesian surrogate models. In *ASME 2006 international design engineering technical conferences and computers and information in engineering conference*, pages 393–404. American Society of Mechanical Engineers Digital Collection, 2006.
- [131] Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383, 2001. doi: 10.1023/A:1012771025575.
- [132] Shengli Xu, Haitao Liu, Xiaofang Wang, and Xiaomo Jiang. A robust error-pursuing sequential sampling approach for global metamodeling based on voronoi diagram and cross validation. *Journal of Mechanical Design*, 136(7):071009, 2014. doi: 10.1115/1.4027161.
- [133] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998. doi: 10.1023/A:1008306431147.
- [134] Karel Crombecq, Dirk Gorissen, Dirk Deschrijver, and Tom Dhaene. A novel hybrid sequential design strategy for global surrogate modeling of computer experiments. *SIAM Journal on Scientific Computing*, 33(4):1948–1974, 2011. doi: 10.1137/090761811.
- [135] John Eason and Selen Cremaschi. Adaptive sequential sampling for surrogate model generation with artificial neural networks. *Computers & Chemical Engineering*, 68:220–232, 2014. doi: 10.1016/j.compchemeng.2014.05.021.
- [136] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996. doi: 10.1613/jair.295.

- [137] Xufeng Yang, Yongshou Liu, Yi Gao, Yishang Zhang, and Zongzhan Gao. An active learning kriging model for hybrid reliability analysis with both random and interval variables. *Structural and Multidisciplinary Optimization*, 51(5):1003–1016, 2015. doi: 10.1007/s00158-014-1189-5.
- [138] Barron J Bichon, Michael S Eldred, Laura Painton Swiler, Sandaran Mahadevan, and John M McFarland. Efficient global reliability analysis for nonlinear implicit performance functions. *AIAA journal*, 46(10):2459–2468, 2008. doi: 10.2514/1.34321.
- [139] B. Echard, N. Gayton, and M. Lemaire. Ak-mcs: An active learning reliability method combining kriging and monte carlo simulation. *Structural Safety*, 33(2):145–154, 2011. ISSN 0167-4730. doi: <https://doi.org/10.1016/j.strusafe.2011.01.002>.
- [140] Zheng Peijuan, Wang Chien Ming, Zong Zhouhong, and Wang Liqi. A new active learning method based on the learning function u of the ak-mcs reliability analysis method. *Engineering Structures*, 148:185–194, 2017. ISSN 0141-0296. doi: <https://doi.org/10.1016/j.engstruct.2017.06.038>.
- [141] Zhen Hu and Sankaran Mahadevan. Global sensitivity analysis-enhanced surrogate (gsas) modeling for reliability analysis. *Structural and Multidisciplinary Optimization*, 53(3):501–521, 2016. doi: 10.1007/s00158-015-1347-4.
- [142] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, page 3149–3157. Curran Associates, Inc., 2017.
- [143] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16*, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785.
- [144] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support, 2018.
- [145] James Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.
- [146] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.
- [147] Max D. Morris and Toby J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43(3):381–402, 1995. ISSN 0378-3758. doi: 10.1016/0378-3758(94)00035-T.
- [148] Max D Morris and Toby J Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43(3):381–402, 1995.
- [149] Ruichen Jin, Wei Chen, and Agus Sudjianto. An efficient algorithm for constructing optimal design of computer experiments. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 37009, pages 545–554, 2003.
- [150] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977. doi: 10.1111/j.2517-6161.1977.tb01600.x.
- [151] Haitao Liu, Shengli Xu, Ying Ma, Xudong Chen, and Xiaofang Wang. An adaptive bayesian sequential sampling approach for global metamodeling. *Journal of Mechanical Design*, 138(1):011404, 2016. doi: 10.1115/1.4031905.

- [152] G. Taguchi and M. S. Phadke. *Quality Engineering through Design Optimization*, pages 77–96. Springer US, Boston, MA, 1989. ISBN 978-1-4684-1472-1. doi: 10.1007/978-1-4684-1472-1_5.
- [153] Berkcan Kapusuzoglu, Yulin Guo, Sankaran Mahadevan, Shunsaku Matsumoto, Miyagi Yoshitomo, Shunsuke Taba, and Daigo Watanabe. Dimension reduction for efficient surrogate modeling in high-dimensional applications. In *AIAA SCITECH 2022 Forum*, page 1440, 2022. doi: 10.2514/6.2022-1440.
- [154] Haitao Liu, Yew-Soon Ong, and Jianfei Cai. A survey of adaptive sampling for global meta-modeling in support of simulation-based complex engineering design. *Structural and Multidisciplinary Optimization*, 57(1):393–416, 2018. doi: 10.1007/s00158-017-1739-8.
- [155] Sankaran Mahadevan, Ruoxue Zhang, and Natasha Smith. Bayesian networks for system reliability reassessment. *Structural Safety*, 23(3):231–251, 2001.
- [156] Chenzhao Li and Sankaran Mahadevan. Role of calibration, validation, and relevance in multi-level uncertainty integration. *Reliability Engineering & System Safety*, 148:32–43, 2016.
- [157] Joshua Mullins and Sankaran Mahadevan. Bayesian uncertainty integration for model calibration, validation, and prediction. *Journal of Verification, Validation and Uncertainty Quantification*, 1(1), 2016. doi: 10.1115/1.4032371.
- [158] *Manufacturing Process Evaluation Under Uncertainty: A Hierarchical Bayesian Network Approach*, volume Volume 1B: 36th Computers and Information in Engineering Conference of International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 08 2016. doi: 10.1115/DETC2016-59226.
- [159] Shankar Sankararaman and Sankaran Mahadevan. Integration of model verification, validation, and calibration for uncertainty quantification in engineering systems. *Reliability Engineering & System Safety*, 138:194–209, 2015. doi: 10.1016/j.res.2015.01.023.
- [160] Zhen Hu and Sankaran Mahadevan. Uncertainty quantification and management in additive manufacturing: current status, needs, and opportunities. *The International Journal of Advanced Manufacturing Technology*, 93(5-8):2855–2874, 2017. doi: 10.1007/s00170-017-0703-5.
- [161] Paul D Arendt, Daniel W Apley, and Wei Chen. Quantification of model uncertainty: Calibration, model discrepancy, and identifiability. *Journal of Mechanical Design*, 134(10):100908, 2012. doi: 10.1115/1.4007390.
- [162] R Timothy Marler and Jasbir S Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004.
- [163] Kais Zaman, Mark McDonald, Sankaran Mahadevan, and Lawrence Green. Robustness-based design optimization under data uncertainty. *Structural and Multidisciplinary Optimization*, 44(2):183–197, 2011. doi: 10.1007/s00158-011-0622-2.
- [164] Xiaoping Du and Wei Chen. Sequential optimization and reliability assessment method for efficient probabilistic design. *Journal of mechanical design*, 126(2):225–233, 2004. doi: 10.1115/1.1649968.
- [165] Darrell Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994. doi: 10.1007/BF00175354.
- [166] Mahmoud H Alrefaei and Sigrún Andradóttir. A simulated annealing algorithm with constant temperature for discrete stochastic optimization. *Management science*, 45(5):748–764, 1999. doi: 10.1287/mnsc.45.5.748.
- [167] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN’95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995. doi: 10.1109/ICNN.1995.488968.

- [168] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017.
- [169] PA Kobryn and SL Semiatin. The laser additive manufacture of ti-6al-4v. *JOM*, 53(9):40–42, 2001. doi: 10.1007/s11837-001-0068-x.
- [170] Jose F Rodriguez, James P Thomas, and John E Renaud. Characterization of the mesostructure of fused-deposition acrylonitrile-butadiene-styrene materials. *Rapid Prototyping Journal*, 6(3):175–186, 2000. doi: 10.1108/13552540010337056.
- [171] Harry A Pierson and Bharat Chivukula. Process–property relationships for fused filament fabrication on preexisting polymer substrates. *Journal of Manufacturing Science and Engineering*, 140(8):084501, 2018.
- [172] Antonio Armillotta, Stefano Bianchi, Marco Cavallaro, and Stefania Minnella. Edge quality in fused deposition modeling: Ii. experimental verification. *Rapid Prototyping Journal*, 2017.
- [173] JM Chacón, Miguel Angel Caminero, Eustaquio García-Plaza, and Pedro J Núñez. Additive manufacturing of pla structures using fused deposition modelling: Effect of process parameters on mechanical properties and their optimal selection. *Materials & Design*, 124:143–157, 2017.
- [174] Ala Qattawi et al. Investigating the effect of fused deposition modeling processing parameters using taguchi design of experiment method. *Journal of Manufacturing Processes*, 36:164–174, 2018.
- [175] SF Costa, FM Duarte, and JA Covas. Thermal conditions affecting heat transfer in fdm/ffe: a contribution towards the numerical modelling of the process: This paper investigates convection, conduction and radiation phenomena in the filament deposition process. *Virtual and Physical Prototyping*, 10(1):35–46, 2015. doi: 10.1080/17452759.2014.984042.
- [176] Wei Chen, Janet K Allen, Kwok-Leung Tsui, and Farrokh Mistree. A procedure for robust design: minimizing variations caused by noise factors and control factors. *Journal of Mechanical Design*, 118(4):478–485, 1996. doi: 10.1115/1.2826915.
- [177] Zhuo Wang, Pengwei Liu, Yaohong Xiao, Xiangyang Cui, Zhen Hu, and Lei Chen. A data-driven approach for process optimization of metallic additive manufacturing under uncertainty. *Journal of Manufacturing Science and Engineering*, 141(8), 2019. doi: 10.1115/1.4043798.
- [178] Lyle Levine, Brandon Lane, Jarred Heigel, Kalman Migler, Mark Stoudt, Thien Phan, Richard Ricker, Maria Strantza, Michael Hill, Fan Zhang, et al. Outcomes and conclusions from the 2018 am-bench measurements, challenge problems, modeling submissions, and conference. *Integrating Materials and Manufacturing Innovation*, 9(1):1–15, 2020. doi: 10.1007/s40192-019-00164-1.
- [179] Brandon Lane, Jarred Heigel, Richard Ricker, Ivan Zhirnov, Vladimir Khromschenko, Jordan Weaver, Thien Phan, Mark Stoudt, Sergey Mekhontsev, and Lyle Levine. Measurements of melt pool geometry and cooling rates of individual laser traces on in625 bare plates. *Integrating Materials and Manufacturing Innovation*, pages 1–15, 2020. doi: 10.1007/s40192-020-00169-1.
- [180] Daniel P Cole, Frank Gardea, Todd C Henry, Jonathan E Seppala, Edward J Garboczi, Kalman D Migler, Christopher M Shumeyko, Jeffrey R Westrich, Sara V Orski, and Jeffrey L Gair. Amb2018-03: Benchmark physical property measurements for material extrusion additive manufacturing of polycarbonate. *Integrating Materials and Manufacturing Innovation*, 9(4):358–375, 2020. doi: 10.1007/s40192-020-00188-y.
- [181] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30 of NIPS’17, page 5580–5590. Curran Associates, Inc., 2017. ISBN 9781510860964.

- [182] Eckart Zitzler, Dimo Brockhoff, and Lothar Thiele. The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In International Conference on Evolutionary Multi-Criterion Optimization, pages 862–876. Springer, 2007. doi: 10.1007/978-3-540-70928-2_64.
- [183] Joshua D Knowles, Lothar Thiele, and Eckart Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. TIK-Report, 214, 2006. doi: 10.3929/ethz-b-000023822.
- [184] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on evolutionary computation, 7(2):117–132, 2003.
- [185] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary computation, 8(2):173–195, 2000. doi: <https://doi.org/10.1162/106365600568202>.
- [186] You Ling and Sankaran Mahadevan. Quantitative model validation techniques: New insights. Reliability Engineering & System Safety, 111:217–231, 2013.
- [187] Ultimaker S5 Specifications. <https://ultimaker.com/3d-printers/ultimaker-s5>, accessed: 02/01/2020.
- [188] Keyence LK-H057 Specifications. <https://www.keyence.com/products/measure/laser-1d/lk-g5000/models/lk-h057/index.jsp>, accessed: 07/29/2019.
- [189] Martin Meckesheimer, Russell R Barton, Timothy W Simpson, and Andrew J Booker. Computationally inexpensive metamodel assessment strategies. In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, volume 80227, pages 191–201. American Society of Mechanical Engineers, 2001. doi: 10.1115/DETC2001/DAC-21028.