METAMODEL-BASED INVERSE UNCERTAINTY QUANTIFICATION OF NUCLEAR REACTOR SIMULATORS UNDER THE BAYESIAN FRAMEWORK

BY

XU WU

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Nuclear, Plasma, and Radiological Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Abstract

Mathematical modeling and computer simulations have long been the central technical topics in practically all branches of science and technology. Tremendous progress has been achieved in revealing quantitative connections between numerical predictions and real-world observations. However, because computer models are reduced representations of the real phenomena, there are always discrepancies between ideal *in silico* designed systems and real-world manufactured ones. As a consequence, uncertainties must be quantified along with the simulation outputs to facilitate optimal design and decision making, ensure robustness, performance or safety margins. Forward uncertainty propagation requires knowledge in the statistical information for computer model random inputs, for example, the mean, variance, Probability Density Functions (PDFs), upper and lower bounds, etc. Historically, "expert judgment" or "user self-evaluation" have been used to specify the uncertainty information associated with random input parameters. Such ad hoc characterization is unscientific and lacks mathematical rigor.

In this thesis, we attempt to solve such "lack of uncertainty information" issue with inverse Uncertainty Quantification (UQ). Inverse UQ is the process to seek statistical descriptions of the random input parameters that are consistent with available high-quality experimental data. We formulate the inverse UQ process under the Bayesian framework using the "model updating equation". Markov Chain Monte Carlo (MCMC) sampling is applied to explore the posterior distributions and generate samples from which we can extract statistical information for the uncertain input parameters. To greatly alleviate the computational burden during MCMC sampling, we used systematically and rigorously developed metamodels based on stochastic spectral techniques and Gaussian Processes (also known as Kriging) emulators.

We demonstrated the developed methodology based on three problems with different levels of sophistication: (1) Point Reactor Kinetics Equation (PRKE) coupled with lumped parameter thermal-hydraulics feedback model based on synthetic experimental data; (2) best-estimate system thermal-hydraulics code TRACE physical model parameters based on OECD/NRC BWR Full-size Fine-Mesh Bundle Tests (BFBT) benchmark steady-state void fraction data; (3) fuel performance code BISON Fission Gas Release (FGR) model based on Risø-AN3 on-line time-dependent FGR measurement data. Metamodels constructed with generalized Polynomial

Chaos Expansion (PCE), Sparse Gird Stochastic Collocation (SGSC) and GP were applied respectively for these three problems to replace the full models during MCMC sampling.

We proposed an improved modular Bayesian approach that can avoid extrapolating the model discrepancy that is learnt from the inverse UQ domain to the validation/prediction domain. The improved approach is organized in a structure such that the posteriors achieved with data in inverse UQ domain is informed by data in the validation domain. Therefore, over-fitting can be avoided while extrapolation is not required. A sequential approach was also developed for test source allocation (TSA) for inverse UQ and validation. This sequential TSA methodology first select tests for validation that has a full coverage of the test domain to avoid extrapolation of model discrepancy term when evaluated at input setting of tests for inverse UQ. Then it select tests that tend to reside in the unfilled zones of the test domain for inverse UQ, so that inverse UQ can extract the most information for posteriors of calibration parameters using only a relatively small number of tests.

The inverse UQ process successfully quantified the uncertainties associated with input parameters that are consistent with the experimental observations. The quantified uncertainties are necessary for future uncertainty and sensitivity study of nuclear reactor simulators in system design and safety analysis. We applied and extended several advanced metamodeling approaches to nuclear engineering practice to greatly reduce the computational cost. The current research bridges the gap between models and data by solving "lack of uncertainty information" issue, as well as providing guidance for improving nuclear reactor simulators through the validation process.

Acknowledgements

First and foremost, this thesis is dedicated to the memory of my father, Jingzeng Wu. He passed away at the beginning of my graduate study and I miss him every day. He offered me plenty of friendly encouragement ever since I started school. I know he is somewhere out there, seeing this process through to its completion. I know it would not be possible without his support and I am very proud to be his son.

I would like to express my sincere gratitude towards my advisor Dr. Tomasz Kozlowski, for giving me the opportunity to work with him, and for his constant guidance and support. Tomasz has been a great mentor and friend to me throughout my graduate study. He never hesitated to offer me opportunities to attend conferences, workshops and summer schools. I would like to thank other professors in my committee: my co-advisor Dr. Hadi Meidani, Dr. Rizwan Uddin and Dr. James Stubbins for their support and insightful discussions. I benefited a lot by attend Dr. Meidani's UQ group meeting.

I would like to thank the entire ARTS group at NPRE for their genuine friendship and encouragement. Dr. Rabie Abu Saleem, Dr. Rijan Shrestha, Guojun Hu, Chen Wang, Daniel O'Grady, Majdi Radaideh, Travis Mui, Guanfeng Gao, Katarzyna Borowiec and many others who have graduated from the group that I cannot name all here. Special thanks go to all the supporting staffs, Becky, Barb, Margaret, as well as those who have retired or left NPRE, Idell, Gail and Rod. I would like to thank them for making our lives easier.

I would like to thank all my friends who shared with me the good times and the bad times through this journey. With a special mention to Dr. Xu Chen, I really enjoyed the time when we bike and work-out together.

I would like to give special thanks to my beloved girlfriend, Cong Xu. We met and fell in love at the beautiful campus at Urbana-Champaign and since summer 2013 we have been maintaining a long-distance relationship as she moved to Penn State University for PhD. She has been offering me great support and comfort during the tough times.

Last but not least, I want to thank my family, who have always been there for me, through tough and happy times. I must express my very profound gratitude to my brother and sister, who have been taking very good care of my mother throughout my years of study in the US. I would not have made this far without their constant love and unfailing support.

Table of contents

List of figures

List of tables

Nomenclatures

AM	Adaptive Metropolis
ANN	Artificial Neural Network
ATHLET	Analysis of THermal-hydraulics of LEaks and Transients
BACCO	Bayesian Analysis of Computer Code Outputs
BC	Boundary Condition
BEPU	Best Estimate plus Uncertainty
BFBT	BWR Full-Size Fine Mesh Bundle Test
BLUE	Best Linear Unbiased Estimator
BLUP	Best Linear Unbiased Predictor
BWR	Boiling Water Reactor
CDF	Cumulative Distribution Function
CI	Credible Intervals
CSAU	Code Scaling Applicability and Uncertainty
CUU	Calibration Under Uncertainty
CV	Cross Validation
DACE	Design and Analysis of Computer Experiments
DAKOTA	Design Analysis Kit for Optimization and Terascale Application
E-M	Expectation-Maximization

FFGP	Function Factorization with Gaussian Process
FGR	Fission Gas Release
GP	Gaussian Processes
GPML	Gaussian Processes for Machine Learning
GRSS	Generalized Residual Sum of Squares
HDMR	High Dimensional Model Representation
HPD	Highest Posterior Density
HTC	Heat Transfer Coefficient
IC	Initial Condition
IETs	Integral effects tests
IFPE	International Fuel Performance Experiments
KDE	Kernel Density Estimation
LBLOCA	Large Break Loss-of-Coolant Accident
LHS	Latin Hypercube Sampling
LOOCV	Leave-One-Out Cross Validation
LWR	Light Water Reactor
M-H	Metropolis-Hastings
M&S	Modeling & Simulation
MAP	Maximum-a-Posteriori
MARS	Multivariate Adaptive Regression Splines
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
MLE	Maximum Likelihood Estimation
MLS	Moving Least-Squares

MMS	Method of Manufactured Solution
MOOSE	Multi-physics Object Oriented Simulation Environment
NISP	Non-Intrusive Spectral Projection
NPP	Nuclear Power Plant
NRC	Nuclear Regulatory Commission
NUPEC	Nuclear Power Engineering Corporation
ODE	Ordinary Differential Equation
OECD	Organisation for Economic Co-operation and Development
OK	Ordinary Kriging
PC	Principal Component
PCA	Principal Component Analysis
PCC	Pearson Correlation Coefficient
PCE	Polynomial Chaos Expansion
PCMI	Pellet-Cladding Mechanical Interaction
PCT	Peak Cladding Temperature
PDE	Partial Differential Equation
PDF	Probability Density Function
PIRT	Phenomena Identification and Ranking Table
PPDF	Posterior Probability Density Function
PRA	Probabilistic Risk Assessment
PRKE	Point Reactor Kinetics Equation
PWR	Pressurized Water Reactor
QoI	Quantity-of-Interest
RBF	Radial Basis Functions

RELAP	Reactor Excursion and Leak Analysis Program
RISMC	Risk Informed Safety Margin Characterization
RSS	Residual Sum of Squares
SA	Sensitivity Analysis
SC	Stochastic Collocation
SETs	Separate Effects Tests
SFEM	Stochastic Finite Element Method
SGSC	Sparse Grid Stochastic Collocation
SIFGRS	Simple Integrated Fission Gas Release and Swelling
SK	Simple Kriging
SQE	Software Quality Engineering
SRCC	Spearman Rank Correlation Coefficient
SRWM	Symmetric Random Walk Metropolis
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TASMANIAN	Toolkit for Adaptive Stochastic Modeling And Non-Intrusive Approximation
TH	Thermal-Hydraulics
TRACE	TRAC/RELAP Advanced Computational Engine
TSA	Test Source Allocation
UK	Universal Kriging
UQ	Uncertainty Quantification
USNRC	U.S. Nuclear Regulatory Commission
V&V	Verification & Validation
VVUQ	Verification, Validation and Uncertainty Quantification

Chapter 1

INTRODUCTION AND LITERATURE SURVEY

1.1 Background and Motivation

During the last four decades, the importance of computer simulations has increased dramatically in furthering our understanding of the responses of engineered systems in real world. Large computer codes that implement complex mathematical models have been successfully applied in the design and performance assessment of real systems in many areas of scientific research. Computer modeling is especially significant to the nuclear engineering community, as physical experimentation is usually too costly or sometimes impossible.

1.1.1 Essential components of computer modeling

To bring up the motivation to perform inverse Uncertainty Quantification (UQ), we first briefly mention some other essential components for computer modeling, including forward UQ:

1. **Verification**: "the process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model" ([103], p. 215). In other words, verification aims to identify, quantify, and reduce errors during the mapping from mathematical model to a computer code.

2. **Code Verification**: the process to access the reliability of the software coding, which includes two activities, numerical algorithm verification and software quality engineering (SQE) [102]. In other words, code verification deals with adequacy of the numerical algorithms and the fidelity of the computer programming to implement these algorithms.

3. **Solution Verification**: also referred to as calculation verification [136], or numerical error estimation [102], is the process to evaluate the numerical accuracy of the solutions to a computer code. The primary difference between code and solution verification is that there is generally no known exact solution to the system of interest for the latter. Solution verification strongly depends on the quality and completeness of code verification, and both processes should be performed prior to validation, as defined below.

4. **Validation**: "the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model" ([103], p. 215). In other words, validation aims to determine the degree of accuracy of the considered model in representing real world phenomena. Verification and Validation together are often termed "**V&V**";

5. **Forward UQ**: the process of quantifying the uncertainties in Quantity-of-Interest (QoIs) by propagating the uncertainties in input parameters through the computer model [126]. QoIs predictions along with uncertainties are necessary for validation;

6. **Sensitivity Analysis** (SA): the study of how uncertainties in the QoIs of can be apportioned to various random input parameters [120];

7. **Optimization**: the process of maximizing or minimizing an object function by systematically choosing input values from within an allowed set [38] [111];

8. **Calibration**: the process of adjusting a set of input parameters implemented in the code so that the agreement of the computer code predictions with corresponding experimental data is maximized [136];

9. **Data Assimilation**: the process to incorporate observations of the actual system into the model state of a numerical model of that system [35]. Data assimilation can be treated as the calibration of dynamic models, which arise in many fields of geosciences such as weather forecasting.

10. **Benchmark**: "A benchmark is a choice of information that is believed to be accurate or true for use in verification, validation or calibration" ([136], p. 1333). For example, benchmarks can be measurements of QoIs from physical experiments or solutions from highly accurate numerical tests;

Figure 1.1 shows the connections between some essential components of computer modeling. From Figure 1.1 it is obvious to see that the forward UQ process always starts with characterization of the input uncertainties, for example, the mean values, variances, Probability

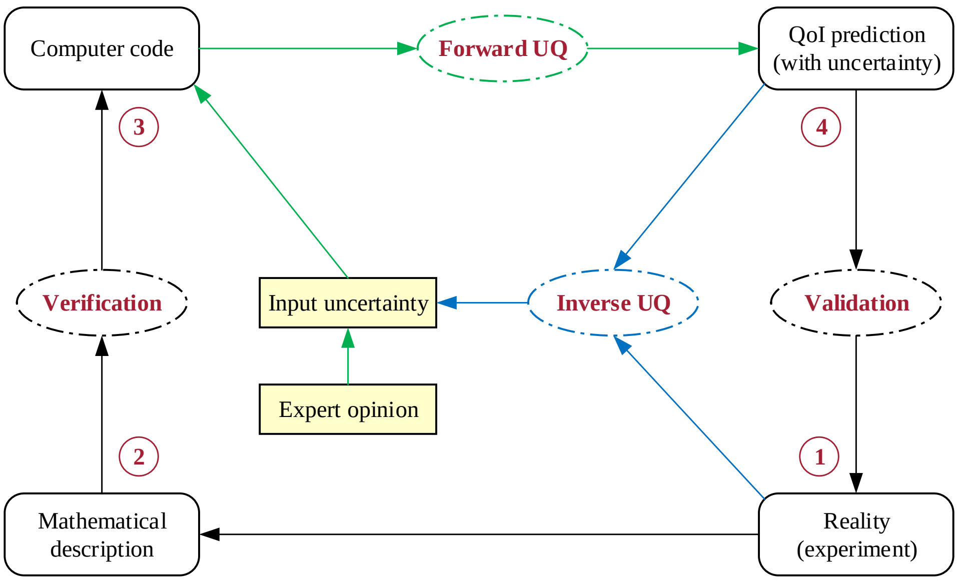


Fig. 1.1 Some essential parts of computer modeling (a non-exclusive list)

Density Functions (PDFs), upper and lower limits, etc. Unfortunately, such information is not always readily available to the code users, which is termed the "**lack of input uncertainty information**" issue. Previously in uncertainty, sensitivity and validation studies of nuclear engineering, "expert opinion" or "user self-assessment" have been widely used, see reviews in [150] [156] [155]. Such ad-hoc specifications of input uncertainty information have been considered reasonable for a long time. However, in many cases they are subjective, unscientific and lacks mathematical rigor. For example, researchers from different institutions may use different choices for the same problem with the same code, resulting in inconsistencies. It is also risky to heavily rely on "expert opinion" because the information we find from previous literature may not always come from "experts".

The "lack of input uncertainty information" issue necessitates the research on inverse UQ. ***Inverse UQ****, also referred to as* ***inverse problem*** *or* ***parameter inference****, is the process to determine the uncertainties in input parameters that characterize the model or code given relevant experimental measurements and code simulations*. Inverse UQ seeks statistical descriptions of the uncertain input model parameters that are consistent with the observed data.

The purpose of this thesis is to replace ad-hoc expert judgment of the statistical properties of input model parameters in nuclear reactor simulation. Bayesian analysis will be used to establish the inverse UQ problems based on experimental data, with systematic and rigorously derived surrogate models.

An early appearance of the term "inverse UQ" can be found in [103], in which it was also termed "backward problem". Some other researchers called it "inverse uncertainty propagation or adjoint method" [137]. According to Oberkampf and Trucano, "The backward problem asks whether we can reduce the output uncertainty by updating the statistical model using comparisons between computations and experiments" ([103], p. 256). Inverse UQ is known to be conceptually and mathematically much more difficult than forward UQ.

1.1.2 The necessity to perform inverse UQ

Mathematical modeling and computer simulations are naturally affected by a relatively large amount of uncertainty in the input data such as model coefficients, forcing terms, boundary conditions, geometry, etc. As a consequence, confidence in modeling & simulation (M&S) must be critically assessed which requires model V&V [102] [103].

The input data uncertainty can be included in the mathematical model adopting a probabilistic setting. In this framework, enough information is required for a complete statistical characterization of the physical system, and the input data is then modelled as random variables. Uncertainties must be quantified along with the simulation outputs to facilitate optimal design and decision making, ensure robustness, performance or safety margins. UQ aims at determining the effect of such input uncertainties on response QoIs and it plays a vital role in the validation process.

M&S are naturally augmented by extensive uncertainty and sensitivity requirements in areas where high efficiency and safety is of critical importance. For example, in nuclear reactor system design where uncertainties must be quantified in order to prove that the investigated design stays within acceptance criteria. In nuclear engineering, UQ and SA are used together to demonstrate that important safety limits are respected with a high confidence level or to identify the main sources of the uncertainties that have to be reduced in order to decrease response variations to acceptable values.

The importance of UQ-supported M&S will continue to grow in the 21th century nuclear engineering, which faces the increasingly stringent demand for risk-informed safety margin characterization and plant performance optimization. This demand is intensified due to the fact that the licensing of nuclear installations has shifted from conservative to **Best Estimate plus Uncertainty (BEPU)** methodologies .

Historically in nuclear system design and assessment, the **conservative** approach has been used to calculate the output response of a code using extreme (unfavorable) values of input parameters. This approach quantifies reactor designs with a considerable margin to assure its safety and avoid under-prediction of safety-relevant parameters (e.g. peak cladding temperature (PCT)) by modeling the physical phenomena such that it always predicts the worst-case scenario . Consequently, as shown in Figure 1.2 the conservative approach leads to considerable inaccuracy in M&S. For example, it consistently over-estimates the PCT and hence under-predicts the time to cladding failure, damaging the economic performance of nuclear systems.

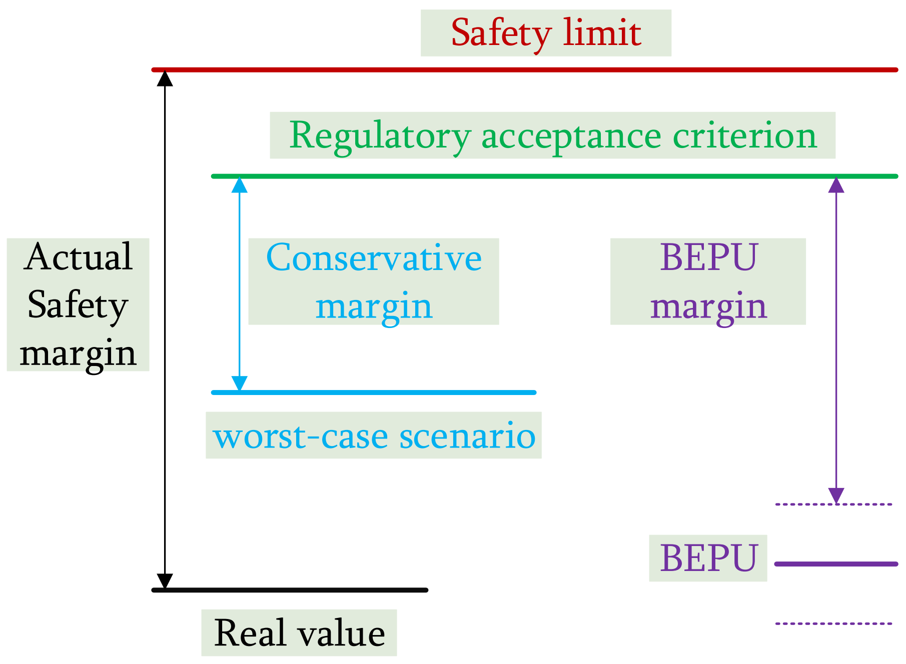


Fig. 1.2 Illustration of conservative and BEPU safety margins

In the 1980s, risk-based safety analysis strategy started to be embedded in the Code Scaling Applicability and Uncertainty (CSAU), and Phenomena Identification and Ranking Table (PIRT) concepts, which was accepted by the U.S. Nuclear Regulatory Commission (USNRC). This strategy is commonly referred to as the BEPU methodology [27] [54] [148]. The goal of BEPU methodologies aims to *capture the physical phenomena as realistically as possible by implementing a wide range of modeling options and increasingly precise calculation methods to capture physical phenomena at a greater fidelity*. According to the BEPU methodology, uncertainties must be quantified in order to prove that the investigated design stays within acceptance criteria.

Validation and thus forward and inverse UQ play a more significant role in nuclear engineering as we are always dealing with high-consequence systems. The decision-making process, development of public policy and preparation of safety procedures all rely on reliable computer codes that have undergone extensive validation process and proven to be of high credibility. Given the limited work on inverse UQ among the nuclear community, we devote this thesis to this topic, hoping to draw more attention into this area and raise the interest on these topics to a higher level in our community.

1.2 Inverse UQ vs. Calibration

Inverse UQ aims to quantify the uncertainty in input parameters such that the discrepancies between code output and observed experimental data can be minimized. Such definition looks very similar with calibration. In this subsection we would like to address the differences between calibration and Inverse UQ. Calibration can be classified as deterministic and statistical calibration [19] [136]. **Deterministic calibration** merely determines the point estimates of best-fit input parameters such that the discrepancies between code output and observed experimental data can be minimized. However, **statistical calibration**, sometimes referred to as **Bayesian calibration** [71], **probabilistic inversion** [139] or **Calibration Under Uncertainty (CUU)** [136], produces statistical descriptions like distributions. In this sense, inverse UQ is same with Bayesian calibration and indeed they do share the same techniques. For example, both of them employ the Bayesian inference theory [44] and explore the posterior PDF with Markov Chain Monte Carlo (MCMC) sampling [50]. They both favour surrogate models when the computational model is expensive. So what makes inverse UQ in the current study different from Bayesian calibration?

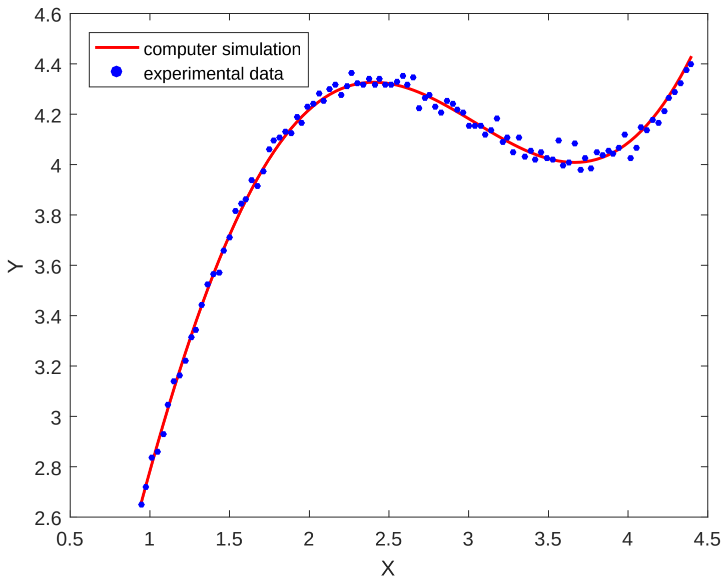


Fig. 1.3 A simple case when calibration won't improve the agreement between simulation and measurement.

Inverse UQ only has very subtle differences with Bayesian calibration. Inverse UQ includes some techniques that implements the Expectation-Maximization (E-M) algorithm [125] rather than sampling of the posterior PDF, even though the former is not as widely applicable as the latter. Bayesian calibration aims at reducing the difference between experiment and simulation, while inverse UQ emphasizes quantifying the input uncertainties. When the model outputs agree very well with experimental data, we may conclude that no calibration is needed. However, the inverse UQ is still useful because the underlying uncertainties in model input parameters have to be quantified. Figure 1.3 illustrates such a case, when the differences between simulation and measurement approximately follow Gaussian noise with a small variance. In that case, calibration is unlikely to improve the agreement between simulation and observation. In essence, in cases where there is no need to do Bayesian calibration, inverse UQ may still be useful. We prefer the term *inverse UQ* to *Bayesian calibration* as it is in agreement with the term *forward UQ*. And it is obvious that they are done in opposite directions.

The advantage of inverse UQ (or Bayesian calibration) over deterministic calibration and "parameter tuning" is apparent. Firstly, information on QoIs from experiments is never sufficiently accurate to allow inference of the "true" or "exact" values of the input parameters. Instead, we can only hope to reduce our ignorance of the parameters by achieving less uncertainties in them (the so-called uncertainty reduction). Secondly, it is highly possible that various combinations of input parameters will yield simulations that have similar agreement with the field data. Deterministic calibration, which relies on optimization techniques to select best-fit values, may end up with getting only one of a set of equally well-fitting values. This is especially true for over-parameterized models given limited data [139]. Thirdly, it is difficult for deterministic calibration to quantify correlations among the estimates. Finally, The observed data usually contains certain degree of uncertainty, which should be considered during the inference process of calibration parameters.

1.3 Literature Survey for Inverse UQ

Very limited work has been one on the topic of inverse UQ in nuclear engineering, or even calibration. Some representative examples will be discussed in the following. Most of them are based on deterministic methods, few of them rely on Bayesian analysis. The application of metamodels during inverse UQ only appears very recently in nuclear engineering.

One of the earlier work on inverse UQ[1] was the "Circé method" presented in [28], which was developed to quantify the uncertainties in the closure laws of Cathare 2 code. The Circé method implemented the E-M algorithm in conjunction with the Maximum Likelihood Estimate (MLE) method. This approach was later extended to a richer mathematical framework includes Maximum a Posteriori (MAP) [125]. In [65], MLE, MAP and MCMC algorithms were used to quantify the uncertainty of two physical models used in TRACE: subcooled boiling heat transfer coefficient (HTC) and interfacial drag coefficient. However, the application of these

[1]Most of the mentioned previous work did not use the term "inverse UQ". But according to our definition they are equivalent to inverse UQ.

methods are limited by several strong assumptions: (1) the relation between the QoI and the input parameter was assumed to be linear; (2) the input parameters were assumed to follow a normal distribution; (3) local sensitivity analysis was required to provide the necessary input for MLE. Furthermore, the performance of this method appears to degenerate when the input dimension increases.

Cacuci and Arslan [17] applied a predictive modeling procedure to reduce the uncertainties in calibration parameters and time-dependent boundary conditions in the large-scale reactor analysis code FLICA4 based on BFBT benchmark, yielding best-estimate predictions of axial void fraction distributions. Bui and co-workers [14] [15] proposed the concept of "total model-data integration" which is based on the theory of Bayesian calibration and a mechanism to accommodate multiple data-streams and models. Such concept allows assimilation of heterogeneous multivariate data in comprehensive calibration and validation of computer models. This approach was demonstrated to the cases of subcooled boiling two-phase flow in nuclear thermal-hydraulics (TH) analysis. Bachoc et al. [7] applied the Bayesian calibration approach to the TH code FLICA 4 in a single-phase friction model. The model discrepancy is modeled with Gaussian Process (GP). This work made inference of the model error for each new potential experimental point, extrapolated from what had been learnt from the available experimental data. The computer code predictive capability was reported to be improved based on the tested case. In another work [142], Bayesian calibration was applied to calibrate the reflowed model parameters in TRACE code. Furthermore, this work considered multivariate time-dependent output and considered the model discrepancy term. However, no results for the model discrepancy term were presented and its extrapolation to the validation and prediction domain was not discussed.

GP emulator-based Bayesian calibration was also applied to fuel performance codes. Higdon et al. [62] used the full Bayesian approach to inversely quantify the uncertainties in four tuning parameters of the FRAPCON code based on fission gas release data from 42 experiments. The measurement uncertainty and the model discrepancy term were quantified simultaneously. Kriging-based calibration was used to quantify the calibration parameters in the fission gas behavior model of fuel performance code BISON [99] [166]. Single experiment was used in [166] while multiple integral experiments were used in [99]. However, they only resulted in an optimized set of fission gas behavior model parameters rather than posterior distributions.

Surrogate-based calibration was also used in other nuclear engineering applications other than TH and fuel performance. Stripling et al. [133] developed a method for calibration and data assimilation using the Bayesian Multivariate Adaptive Regression Splines (MARS) emulator as a surrogate for the computer code. Their method started with sampling of the uncertain input space. The emulator was then used to assign weights to the samples which were applied to produce the posterior distributions of the inputs. This approach was applied to the calibration of a Hyades 2D model of laser energy deposition in beryllium. The major difference of this approach with MCMC-based Bayesian calibration is that, it generates samples beforehand and the candidate acceptance routine in MCMC sampling is replaced with a weighting scheme. Note that such approach does not include a model discrepancy term. Yurko et al. [167] used the Function Factorization with Gaussian Process (FFGP) priors model to emulate the behavior of computer codes. Calibration of a simple friction-factor example using a Method of Manufactured Solution (MMS) with synthetic observational data was used to demonstrate the key properties of this method. This approach is better suited for the emulation of complex time series outputs.

The investigation of inverse problems has a longer history in other areas than nuclear engineering, resulting in abundant collection of references, see reviews in [39] [134]. Previous research on inverse UQ problems primarily lie in the context of source inversion and calibration of model input parameters. Some representative examples are identification of source term and deposition velocity in nuclear radiation release [71], source inversion in transient diffusion [88] [89], source inversion in heat conduction and permeability estimation in flow through porous media [82], recovery of the location and intensity of a radiation source in the urban area [129]. Other applications can be found for simply supported beam [4], charged particle accelerator and spot welding experiment [63], shock physics, materials science, cosmology, and particle physics [61], thermally decomposing foam [91], process-based forest models [139] and climate model with terrestrial carbon cycle [144], etc.

1.4 Outline of the Thesis

This thesis is organized in the following wa. Chapter 2 will introduce the formulation of inverse UQ problem under the Bayesian framework, full and modular Bayesian approaches, an improved modular Bayesian approach and MCMC sampling techniques. Chapter 3 will provide details of stochastic spectral methods. Chapter 4 will demonstrate how to use GP to build emulators for deterministic computer models. We will demonstrate the established inverse UQ methodology for several problems of different level of sophistication in Chapter 5 - 8. Chapter 9 summarizes this thesis and provides a discussion on future work.

Chapter 2

INVERSE UQ FORMULATION

In this chapter, the general theory for inverse UQ within the Bayesian framework is introduced. In Section 2.1, the formulation of the inverse UQ problem for a general computer model is provided. In Section 2.2, the full Bayesian and modular Bayesian approaches are introduced, compared and their limitations are explained. In Section 2.3, we propose an improved modular Bayesian approach that is capable of accounting for model discrepancy, as well as avoiding the extrapolation issue associated with full and modular Bayesian approaches. In Section 2.4, a few adaptive MCMC sampling algorithms are presented to efficiently explore the posterior PDF.

2.1 Inverse UQ Problem Formulation

In this Section, We start with a classification of the inputs parameters. Then the “model updating equation" is introduced that incorporates the model discrepancy term. Finally, the Bayesian solution for the inverse UQ process is included.

2.1.1 Classification of input parameters

Consider a general computer model $\mathbf{y}^{\mathrm{M}} = \mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta})$ where $\mathbf{y}^{\mathrm{M}}$ is the model output (also called response or QoI) which can be either a scalar or vector that corresponds to multi-dimensional outputs. The vector $\mathbf{x} = [x_1, x_2, \dots, x_r]^\top$ is the vector of **design variables** (also called **system inputs**, **control variables**, or **observable variables**), and $\boldsymbol{\theta} = [\theta_1, \theta_2, \dots, \theta_d]^\top$ is the vector of **calibration parameters** (sometimes called **ancillary variables**). Examples of design variables are initial conditions (ICs) and boundary conditions (BCs). Calibration parameters are specified as inputs to the computer model but are unknown or not measurable when conducting the physical experiments. In this work we use a broad definition of calibration parameters similar to [144], which include:

1. physical constants, such as physical model parameters like material properties and heat transfer coefficients (HTC);
2. tuning parameters, which are needed to make the model perform well, like multiplicative or additive factors. Tuning parameters are usually notional and of little or no physically interpretable meaning.
3. context-specific constants, such as switch between different scenarios. For example, switch between various flow regimes in nuclear reactor system thermal-hydraulics analysis.

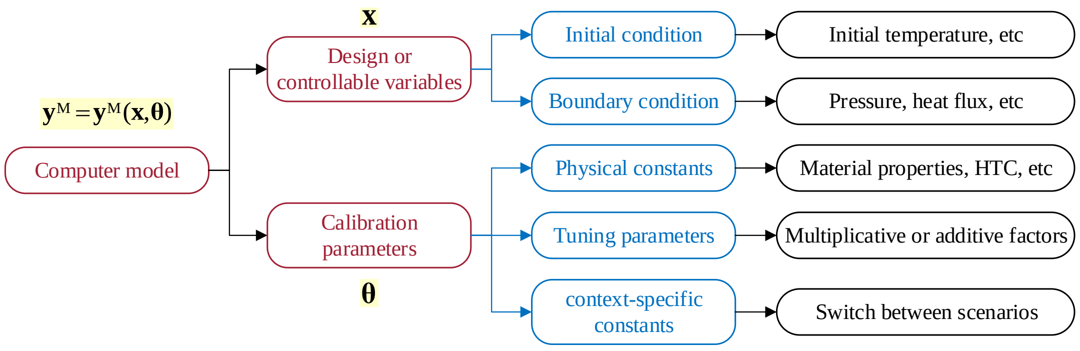


Fig. 2.1 Classification of input parameters for a general computer model.

Figure 2.1 shows the classification of input parameters for a general computer model. We would like to point out a few distinctions between $\mathbf{x}$ and $\boldsymbol{\theta}$ as it is important to avoid the confusions before inverse UQ:

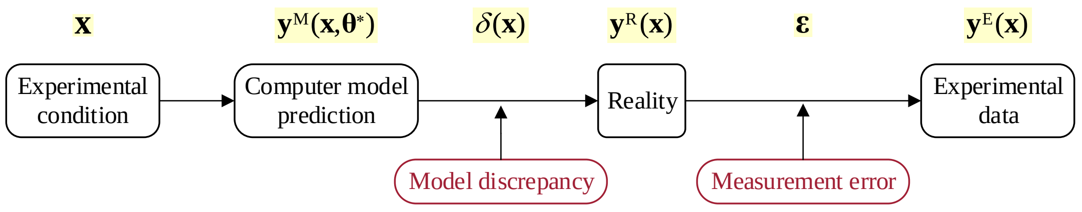
1. Design variables are usually required by both computer simulation and physical experimentation, while calibration parameters are only needed by the former.
2. Design variables usually have clear and unambiguous physical meaning, while calibration parameters may have a physical meaning in nature or be purely numerical.
3. Design variables are used to describe scenarios/experiments that have been observed or to be performed, while the calibration parameters are assumed to have “real" or “true" values that remain unchanged across all scenarios of interest [19].
4. Design variables are usually assumed to be known or at least observable during experimentation. They can also subject to uncertainties due to “variability", which are assumed to be reported along with the benchmark. Calibration parameters, however, have unknown uncertainties and are usually characterized by prior or posterior distributions representing epistemic uncertainty rather than variability.

5. The distinction between $\mathbf{x}$ and $\boldsymbol{\theta}$ is not important for many purposes like forward UQ and sensitivity analysis. But in the context of inverse UQ, calibration parameters are the only targets.

It has to be noted that some different definitions or classifications on models inputs have appeared in previous work on calibration, validation or model updating. In the work of Campbell [19], “inputs” and “parameters” are used to represent design variables $\mathbf{x}$ and calibration parameters $\boldsymbol{\theta}$ respectively. However, this can easily cause confusion for the readers. Therefore, we assume no difference between “input", “variable” and “parameter” and will use “design” and “calibration” in front of these terms to explicitly refer to $\mathbf{x}$ and $\boldsymbol{\theta}$. In the work of Kennedy and O’Hagan [71], design variables $\mathbf{x}$ are also called “variable inputs". We referred the readers to their work for a more complete taxonomy of the uncertainties involved in computer models.

2.1.2 Model updating equation

We use “experiment”, “observation” and “measurement” interchangeably in this work, assuming no difference between them and also use the terms “physical” or “field” in front of them in order to be consistent with the open literature. The connections between computer model simulation, reality and experiments are illustrated in Figure 2.2. Given an experimental condition characterized by design variables $\mathbf{x}$, to learn about the reality or true value of the QoIs $\mathbf{y}^{\mathrm{R}}(\mathbf{x})$, we run the computer model to get $\mathbf{y}^{\mathrm{M}}(\mathbf{x},\boldsymbol{\theta}^*)$. We use $\boldsymbol{\theta}^*$ to represent the “best” or “true[1]” but unknown values for the calibration parameters, the learning of which is the goal of inverse UQ process.



$\mathbf{x}$
$\mathbf{y}^{\mathrm{M}}(\mathbf{x},\boldsymbol{\theta}^*)$
$\delta(\mathbf{x})$
$\mathbf{y}^{\mathrm{R}}(\mathbf{x})$
$\boldsymbol{\varepsilon}$
$\mathbf{y}^{\mathrm{E}}(\mathbf{x})$
Experimental condition
Computer model prediction
Reality
Experimental data
Model discrepancy
Measurement error

Fig. 2.2 The connections between computer model prediction, reality and experimental data.

Because computer models are always reduced rep 2.2resentations of the reality, we introduce the term $\delta(\mathbf{x})$ to represent the discrepancy between computer simulation and reality. $\delta(\mathbf{x})$ is the **model uncertainty**, also called **model discrepancy**, **model inadequacy** or **model bias/error**

[1]By “best” we mean that the model run at $\boldsymbol{\theta}^*$ gives the most accurate prediction. However, it has to be noted that the “best” value may be different from the “real” value. $\boldsymbol{\theta}^*$ is the “best” value only in the sense of most accurately representing the measurement data. Due to model discrepancy, model prediction may not agree well with reality when the model runs at the “real” value. However, since the “real” value can never be learnt, by convention the “best” value and “real” value are treated as the same.

[4] [13] [71]. Model discrepancy $\delta(\mathbf{x})$ is due to incomplete or inaccurate underlying physics, numerical approximation errors, and/or other inaccuracies that would exist even if all the parameters in the computer model were known. We now have the following relation:

$$\mathbf{y}^{\mathrm{R}}(\mathbf{x}) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta}^{*}) + \delta(\mathbf{x}) \tag{2.1}$$

To learn the reality $\mathbf{y}^{\mathrm{R}}(\mathbf{x})$, we also perform experiments to get observation $\mathbf{y}^{\mathrm{E}}(\mathbf{x})$. In the experimental process we will inevitably have measurement error/noise:

$$\mathbf{y}^{\mathrm{E}}(\mathbf{x}) = \mathbf{y}^{\mathrm{R}}(\mathbf{x}) + \boldsymbol{\varepsilon} \tag{2.2}$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}})$ represents the **measurement error**. Note that there can be multiple measurements and it is widely accepted to have homoscedastic experimental errors $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}} = \sigma_{\varepsilon}^{2}\mathbf{I}$. Also, $\boldsymbol{\mu} = \mathbf{0}$ is frequently used, assuming that the instrumentation has no systematic bias and the mean value of the measurement is same with reality.

The model discrepancy term was first addressed in the seminal work of Kennedy and O'Hagan [71]. It is important to consider model discrepancy $\delta(\mathbf{x})$ as otherwise we would have an unrealistic level of confidence in the computer model predictions [13]. Equation 2.1 shows that without $\delta(\mathbf{x})$ we will have "model = reality", which is not reasonable and will cause "over-fitting". Over-fitting means that the calibration parameters are so over-calibrated according to certain set of experiments that the computer code may perform poorly when applied to other experiments. In the TRACE application we will demonstrate that the introduction of the model discrepancy term into the methodology presented in Section 2.3 is capable of avoid over-fitting. By combining Equations 2.1 and 2.2 we have:

$$\mathbf{y}^{\mathrm{E}}(\mathbf{x}) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta}^{*}) + \delta(\mathbf{x}) + \boldsymbol{\varepsilon} \tag{2.3}$$

Equation 2.3 is frequently referred to as "model updating formulation/equation" [4]. The model updating equation will serve as the starting point of the inverse UQ process. In the work of Kennedy and O'Hagan [71], they also used a similar equation. However, they assigned an unknown regression parameter ρ along with the model output $\mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta}^{*})$. In this work, we choose not to use such a parameter. In the next subsection, we will use Bayesian inference and model updating equation to formulate the posterior distributions for the "true" values $\boldsymbol{\theta}^{*}$ of the calibration parameters.

2.1.3 Bayesian solution for inverse UQ problem

The inverse UQ process seeks input values that are most consistent with the measurement data, the computer model and any prior beliefs gained through previous experiments or expert judgment. The Bayesian inference theory [44] is used to find the posterior PDF of the "best" input parameters $\boldsymbol{\theta}^*$ which is defined as $p\left(\boldsymbol{\theta}^*|\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}\right)$, where $\mathbf{y}^{\mathrm{E}}$ and $\mathbf{y}^{\mathrm{M}}$ are the ensembles of field observations and computer simulations respectively. According to the Bayesian theory we have:

$$p\left(\boldsymbol{\theta}^*|\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}\right) \propto p\left(\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}|\boldsymbol{\theta}^*\right) \cdot p\left(\boldsymbol{\theta}^*\right) \tag{2.4}$$

where $p\left(\boldsymbol{\theta}^*\right)$ is the prior and $p\left(\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}|\boldsymbol{\theta}^*\right)$ is the likelihood function. In brief, prior and posterior probabilities represent degrees of belief about possible values of $\boldsymbol{\theta}^*$, before and after observing the experimental data $\mathbf{y}^{\mathrm{E}}$. Given a prior for $\boldsymbol{\theta}^*$, the likelihood function measures the probability of the observed data $\mathbf{y}^{\mathrm{E}}$ being drawn from it. Once we have a specification for the likelihood function, finding the posterior PDF is essentially just an integral calculation which can be done by using techniques like MCMC [50]. Analytical solutions of the posterior may be also possible in rare conditions when the prior and likelihood are sufficiently simple. The formulation of the likelihood function $p\left(\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}|\boldsymbol{\theta}^*\right)$ is a challenging task as we need proper definitions for model discrepancy.

We have previously treated the measurement error $\boldsymbol{\varepsilon}$ as i.i.d. zero-mean Gaussian noise, whose variance is expected to be reported along with measurement data since the error rates for most instrumentation are known. The model discrepancy $\delta(\mathbf{x})$ needs a proper formulation. This is still an area of active research [79]. GP with uninformative prior on any parameters in $\delta(\mathbf{x})$ is a popular choice [61] [63] [71] [144]. However, such formulation is usually problem specific [144]. A more detailed and complete discussion of the model discrepancy term will be presented in Section 2.2 where we introduce the modular Bayesian approach.

Besides a proper formulation, another important issue associated with model discrepancy $\delta(\mathbf{x})$ is called the "identifiability" [23]. Identifiability answers the question that whether the "true" value $\boldsymbol{\theta}^*$ can theoretically be inferred based on the available measurement data, as it is difficult to know how much of the difference between computer output and field experiments should be attributed to the uncertainty of $\boldsymbol{\theta}^*$, model discrepancy $\delta(\mathbf{x})$ and measurement error $\boldsymbol{\varepsilon}$ respectively. In other words, different combinations of uncertainties caused by $\boldsymbol{\theta}^*$, $\delta(\mathbf{x})$ and $\boldsymbol{\varepsilon}$ can account for the same distinction between model prediction and field measurements, making the "true" value $\boldsymbol{\theta}^*$ not identifiable.

Given the above discussion, $\boldsymbol{\varepsilon} = \mathbf{y}^{\mathrm{E}}(\mathbf{x}) - \mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta}^*) - \delta(\mathbf{x})$ follows a multi-dimensional Gaussian distribution. The posterior can be written as:

$$p\left(\boldsymbol{\theta}^* | \mathbf{y}^{\mathrm{E}}, \mathbf{y}^{\mathrm{M}}\right) \propto p\left(\boldsymbol{\theta}^*\right) \cdot \frac{1}{\sqrt{|\boldsymbol{\Sigma}|}} \exp\left[-\frac{1}{2}\left[\mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}} - \delta\right]^{\top} \boldsymbol{\Sigma}^{-1}\left[\mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}} - \delta\right]\right] \tag{2.5}$$

If heteroscedastic experimental errors are assumed between different measurements or different QoIs, we should use $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}})$ where $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}$ is the covariance matrix for measurement error. The diagonal entries represent the variances for each error component while the off-diagonal elements are their covariance. Note that $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}$ is only part of the likelihood covariance matrix $\boldsymbol{\Sigma}$, which includes other sources of uncertainties as shown in Figure 2.3.

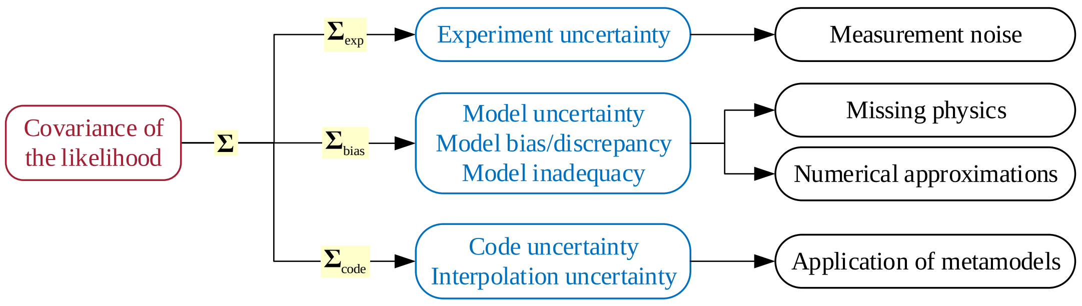

Covariance of the likelihood
$\boldsymbol{\Sigma}$
$\boldsymbol{\Sigma}_{\text{exp}}$
$\boldsymbol{\Sigma}_{\text{bias}}$
$\boldsymbol{\Sigma}_{\text{code}}$
Experiment uncertainty
Model uncertainty
Model bias/discrepancy
Model inadequacy
Code uncertainty
Interpolation uncertainty
Measurement noise
Missing physics
Numerical approximations
Application of metamodels

Fig. 2.3 Components of the covariance matrix of the likelihood function.

From Figure 2.3, We have:

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_{\text{exp}} + \boldsymbol{\Sigma}_{\text{bias}} + \boldsymbol{\Sigma}_{\text{code}} \tag{2.6}$$

where $\boldsymbol{\Sigma}_{\text{exp}} = \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}$ is the **experimental uncertainty** caused by measurement noise. The second term $\boldsymbol{\Sigma}_{\text{bias}}$ represents the **model uncertainty** due to, as we have discussed previously, incomplete or inaccurate underlying physics and numerical approximation errors. The third term $\boldsymbol{\Sigma}_{\text{code}}$ is called **code uncertainty**, or **interpolation uncertainty**, because we do not know the computer code outputs at every input, especially when the code is computationally prohibitive. In this case, we might choose to use some kind of metamodels. For example, GP is a very good choice for metamodels as it provide an estimation of the code uncertainty $\boldsymbol{\Sigma}_{\text{code}}$.

However, in practical application is not always possible to include the model discrepancy term in the analysis. In those cases we have to neglect $\delta(\mathbf{x})$, and the posterior in Figure 2.5 becomes Equation 2.7. Neglecting the model discrepancy will cause over-fitting as we have commented before.

$$p\left(\boldsymbol{\theta}^* | \mathbf{y}^{\mathrm{E}}, \mathbf{y}^{\mathrm{M}}\right) \propto \cdot \frac{p\left(\boldsymbol{\theta}^*\right)}{\sqrt{|\boldsymbol{\Sigma}_{\text{exp}} + \boldsymbol{\Sigma}_{\text{code}}|}} \exp\left[-\frac{1}{2}\left[\mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}}\right]^{\top}\left(\boldsymbol{\Sigma}_{\text{exp}} + \boldsymbol{\Sigma}_{\text{code}}\right)^{-1}\left[\mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}}\right]\right] \tag{2.7}$$

2.1.4 Motivation for using metamodels

The posterior PDF $p\left(\boldsymbol{\theta}^*|\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}\right)$ is the Bayesian solution to the inverse UQ problem. Posterior PDF represents the uncertainty about $\boldsymbol{\theta}^*$ informed by given experimental data. Various statistical moments and probability densities can be computed as long as we manage to generate samples that the follow posterior PDF $p\left(\boldsymbol{\theta}^*|\mathbf{y}^{\mathrm{E}},\mathbf{y}^{\mathrm{M}}\right)$. The general idea is to generate samples that follow a probability density which is approximately equal to the posterior PDF without knowing the normalizing constant. In Section 2.4 we will introduce a few efficient adaptive MCMC sampling techniques. The most important issue with MCMC sampling is that tens of thousands of MCMC samples are usually required to sufficiently explore the posterior PDF, which poses challenges for computationally prohibitive codes as each MCMC sample requires a full model execution. This issue is frequently bypassed by using metamodels.

Metamodels are approximations of the input/output relation of a computer model. They are also called **surrogate models**, **response surfaces** or **emulators**[2] . They are built from a limited number of runs of the full model at specially selected values of the random input parameters (the so-called experimental design [119] [121]) and a learning algorithm. Metamodels usually take much less computational time than the full model while maintaining the input/output relation to a desirable accuracy. Once validated, metamodels can be used to perform uncertainty and sensitivity analysis, validation, optimization, etc., for which the other available methods can incur an excessive computational burden as they require hundreds or thousands of computer model simulations.

A computer model is already a reduced representation or an abstraction of phenomena in the real world. A metamodel is yet another approximation of a computer model, highlighting properties of the model itself. Therefore, no real physics is considered during the construction of a metamodel. Metamodels are widely used in Verification, Validation and Uncertainty Quantification (VVUQ) and optimization study. However, when it comes to explain the real world phenomena, the computer model should be used instead of the metamodel.

Typically examples of metamodels are low-order polynomial regression, Moving Least-Squares (MLS), Radial Basis Functions (RBF), Artificial Neural Network (ANN), Support Vector Machine (SVM), Gaussian Process (GP, also called Kriging), generalized Polynomial Chaos Expansion (PCE), Sparse Grid Stochastic Collocation (SGSC), etc. See [38] [111] for detailed reviews of metamodels. Previously, metamodels built with stochastic spectral methods like PCE and SGSC [82] [87] [88] [89] [149] [150] [152] [156] [158] and GP emulators [7]

[2]The term "emulator" (as a comparison, the original or full model is called a "simulator") is often used for probabilistic response surfaces whose estimation at an untried input is a distribution rather than a point value. Therefore an emulator is a statistical approximation of the simulator.

[61] [63] [71] [91] [104] [139] [144] [153] [155] [166] [167] are especially popular and have been widely used in Bayesian inference/calibration.

2.2 Full vs. Modular Bayesian Approach

In this section, we introduce the full and modular Bayesian approaches that have been used in the literature for inverse UQ. The majority of their implementations follow the seminal work of Kennedy and O'Hagan [71], hereafter referred to as the "KOH" method. The KOH method became part of the later developed framework called Bayesian Analysis of Computer Code Outputs (BACCO) [104]. Research in DACE (Design and Analysis of Computer Experiments) [119] [121] was the forerunner of BACCO. The BACCO approach uses GP to build emulators for complex computer models, and the use these emulators to perform calibration, uncertainty and sensitivity analysis.

Bayesian analysis simultaneously incorporate all relevant information and deals with all uncertainties in the model. In full Bayesian approach, all the unknown hyperparameters in the GP emulators for both the computer model and its discrepancy term are treated in a similar way as the calibration parameters. They are assigned priors, enter the likelihood function and eventually their posteriors are solved simultaneously. Full Bayesian analysis results in an extremely complicated function for the posterior of calibration parameters and GP hyperparameters. Then the GP hyperparameters need to be integrated out from the joint posterior to get marginal distributions of the calibration parameters [9]. However, in modular Bayesian approach, the estimation of calibration parameters, GP hyperparameters for computer model and model discrepancy are all separated. The details of both approaches will be described in the following subsections. We will use $\boldsymbol{\theta}$ instead of $\boldsymbol{\theta}^*$ to represent the target of inverse UQ. In Bayesian analysis, all unknowns are considered random. Therefore, we drop the superscript for notational simplicity.

2.2.1 Full Bayesian approach

The key components of full Bayesian approach are illustrated in Figure 2.4. Given experimental observations, a test source allocation (TSA) algorithm is first implemented to separate them for inverse UQ $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}})$ and validation $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}})$ because the same data should not be used for both purposes. Next two GP emulators are built for the computer model and its discrepancy term:

$$\mathbf{y}^{\mathrm{M}}(\mathbf{x},\boldsymbol{\theta}) \sim \mathcal{GP}\left\{\left(\mathbf{f}^{\mathrm{M}}(\mathbf{x},\boldsymbol{\theta})\right)^{\top}\boldsymbol{\beta}^{\mathrm{M}}, \sigma_{\mathrm{M}}^{2}\mathcal{R}^{\mathrm{M}}\left\langle\left(\mathbf{x}^{(i)},\boldsymbol{\theta}^{(i)}\right),\left(\mathbf{x}^{(j)},\boldsymbol{\theta}^{(j)}\right)\right\rangle\right\} \tag{2.8}$$

$$\delta(\mathbf{x}) \sim \mathcal{GP}\left\{\left(\mathbf{f}^{\delta}(\mathbf{x})\right)^{\top}\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}\mathcal{R}^{\delta}\left\langle \mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right\rangle\right\} \tag{2.9}$$

Note that the first GP emulator takes both $\mathbf{x}$ and $\boldsymbol{\theta}$ as inputs, while the second GP emulator only takes $\mathbf{x}$. The basis functions $\mathbf{f}^{\mathrm{M}}$ and $\mathbf{f}^{\delta}$ are chosen by the user. The hyperparameters for the computer model $\boldsymbol{\Psi}^{\mathrm{M}} = \left\{\boldsymbol{\beta}^{\mathrm{M}}, \sigma_{\mathrm{M}}^{2}, \boldsymbol{\theta}^{\mathrm{M}}, \mathbf{p}^{\mathrm{M}}\right\}$ and the model discrepancy $\boldsymbol{\Psi}^{\delta} = \left\{\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}, \boldsymbol{\theta}^{\delta}, \mathbf{p}^{\delta}\right\}$ are unknown. Replacing $\mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta})$ and $\delta(\mathbf{x})$ in the "model updating equation" we get:

$$\mathbf{y}^{\mathrm{E}} = \mathcal{GP}\left\{\left(\mathbf{f}^{\mathrm{M}}\right)^{\top}\boldsymbol{\beta}^{\mathrm{M}}, \sigma_{\mathrm{M}}^{2}\mathcal{R}^{\mathrm{M}}\right\} + \mathcal{GP}\left\{\left(\mathbf{f}^{\delta}\right)^{\top}\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}\mathcal{R}^{\delta}\right\} + \boldsymbol{\varepsilon} \tag{2.10}$$

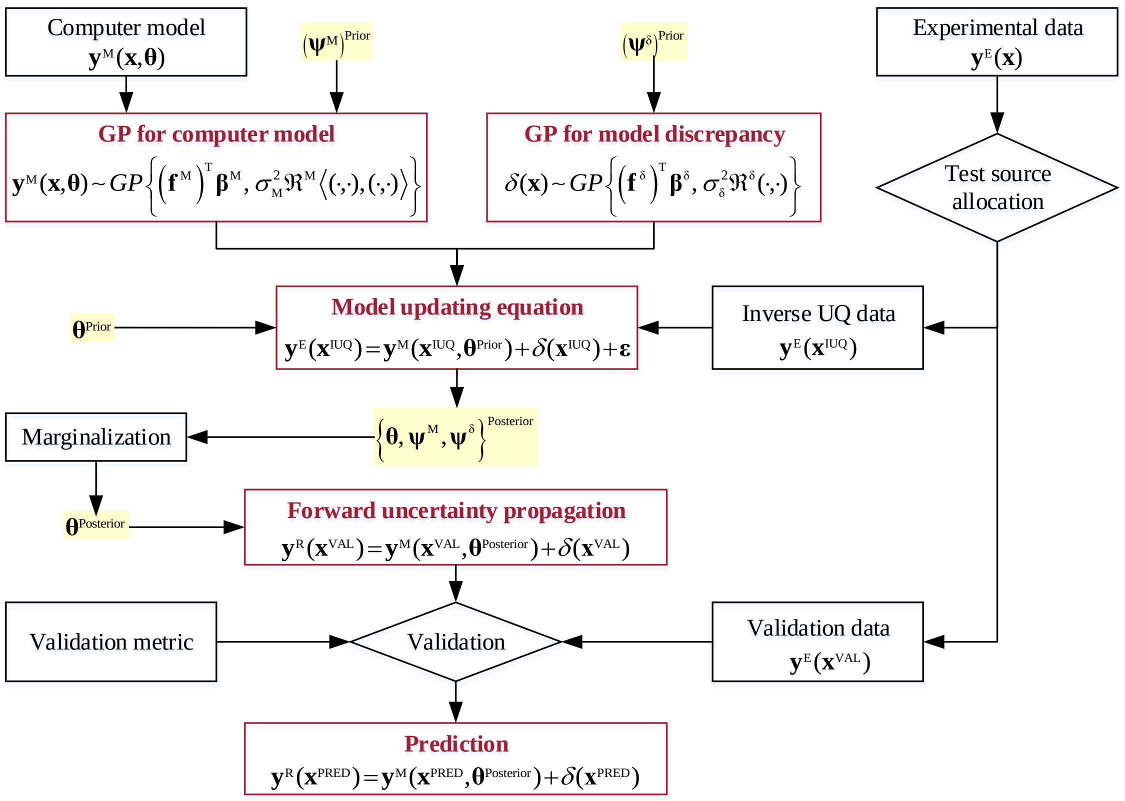

Computer model $\mathbf{y}^{\mathrm{M}}(\mathbf{x},\boldsymbol{\theta})$
$(\boldsymbol{\Psi}^{\mathrm{M}})^{\mathrm{Prior}}$
$(\boldsymbol{\Psi}^{\delta})^{\mathrm{Prior}}$
Experimental data $\mathbf{y}^{\mathrm{E}}(\mathbf{x})$
GP for computer model
$\mathbf{y}^{\mathrm{M}}(\mathbf{x},\boldsymbol{\theta}) \sim GP\left\{\left(\mathbf{f}^{\mathrm{M}}\right)^{\mathrm{T}}\boldsymbol{\beta}^{\mathrm{M}}, \sigma_{\mathrm{M}}^{2}\Re^{\mathrm{M}}\langle(\cdot,\cdot),(\cdot,\cdot)\rangle\right\}$
GP for model discrepancy
$\delta(\mathbf{x}) \sim GP\left\{\left(\mathbf{f}^{\delta}\right)^{\mathrm{T}}\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}\Re^{\delta}(\cdot,\cdot)\right\}$
Test source allocation
$\boldsymbol{\theta}^{\mathrm{Prior}}$
Model updating equation
$\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}}) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{IUQ}},\boldsymbol{\theta}^{\mathrm{Prior}}) + \delta(\mathbf{x}^{\mathrm{IUQ}}) + \boldsymbol{\varepsilon}$
Inverse UQ data $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}})$
Marginalization
$\{\boldsymbol{\theta}, \boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\}^{\mathrm{Posterior}}$
$\boldsymbol{\theta}^{\mathrm{Posterior}}$
Forward uncertainty propagation
$\mathbf{y}^{\mathrm{R}}(\mathbf{x}^{\mathrm{VAL}}) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}},\boldsymbol{\theta}^{\mathrm{Posterior}}) + \delta(\mathbf{x}^{\mathrm{VAL}})$
Validation metric
Validation
Validation data $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}})$
Prediction
$\mathbf{y}^{\mathrm{R}}(\mathbf{x}^{\mathrm{PRED}}) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{PRED}},\boldsymbol{\theta}^{\mathrm{Posterior}}) + \delta(\mathbf{x}^{\mathrm{PRED}})$

Fig. 2.4 Workflow of the full Bayesian approach.

Following the KOH approach, the computer model $\mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta})$, model discrepancy $\delta(\mathbf{x})$ and measurement error $\boldsymbol{\varepsilon}$ are assumed independent. Such assumption has been adopted in all the previous work on Bayesian calibration and significantly simplifies the calculations. Full Bayesian approach then proceeds with assigning prior distributions for all the unknowns shown below:

$$\left\{\boldsymbol{\theta}, \boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}, \sigma_{\varepsilon}^{2}\right\} = \left\{\boldsymbol{\theta}, \boldsymbol{\beta}^{\mathrm{M}}, \sigma_{\mathrm{M}}^{2}, \boldsymbol{\theta}^{\mathrm{M}}, \mathbf{p}^{\mathrm{M}}, \boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}, \boldsymbol{\theta}^{\delta}, \mathbf{p}^{\delta}, \sigma_{\varepsilon}^{2}\right\} \tag{2.11}$$

Note that the measurement error σ_{ε}^{2} is sometimes treated as known [144] because the error rates for most instrumentation tools are known. We assume that it will usually be reported with the benchmark data. If the σ_{ε}^{2} is unknown, it can be learnt simultaneously with the calibration parameters and GP hyperparameters [10] [63] [61]. Furthermore, upon selection of a correlation kernel, the roughness parameters $\mathbf{p}^{\mathrm{M}}$ and $\mathbf{p}^{\delta}$ are usually known. However, even after assuming they are known, we are still left with many parameters to evaluate $\left\{\boldsymbol{\theta}, \boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\right\} = \left\{\boldsymbol{\theta}, \boldsymbol{\beta}^{\mathrm{M}}, \sigma_{\mathrm{M}}^{2}, \boldsymbol{\theta}^{\mathrm{M}}, \boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}, \boldsymbol{\theta}^{\delta}\right\}$. There is no guidance for the selection of proper prior distributions for them, especially the hyperparameters. Previously, researchers have been using distributions like normal, gamma, beta and inverse gamma [24][25][31], but there are no explanations about how the distribution parameters were chosen.

Next a joint likelihood function is formulated for all the unknowns $\{\boldsymbol{\theta}, \boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\}$. Given the inverse UQ measurement data $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}})$ and computer model training samples $\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{IUQ}}, \boldsymbol{\Theta})$ where Θ represents the training domain, MCMC sampling can be used to generate samples for the unknowns. In this way, full Bayesian analysis achieves the uncertainties in calibration parameters and model discrepancy term simultaneously. Marginalization is required to integrate out $\{\boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\}$ to get $\boldsymbol{\theta}^{\mathrm{Posterior}}$. After inverse UQ, $\boldsymbol{\theta}^{\mathrm{Posterior}}$ can be directly used for validation and prediction. Validation and prediction are not the focus in this thesis, they are mentioned only to complete the introduction of the workflow of full Bayesian analysis shown in Figure 2.4.

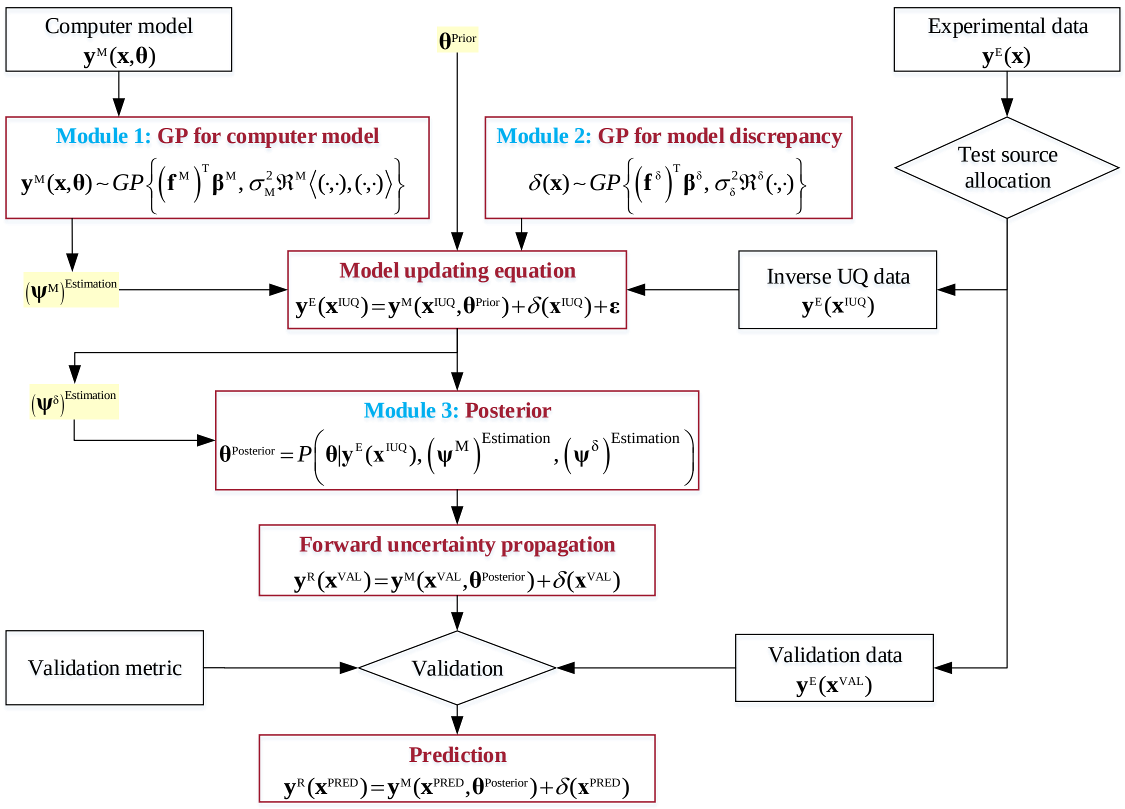
2.2.2 Modular Bayesian approach

Modularization is a technique to separate various modules in Bayesian analysis to prevent suspect information belonging to one part from overly influencing another part [80]. The most important difference of modular Bayesian with full Bayesian is that the former separates the estimation of $\{\boldsymbol{\theta}, \boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\}$. Modular Bayesian uses plausible estimates (e.g. MLEs) of $\{\boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\}$ and treat them as if they were the true values of $\{\boldsymbol{\Psi}^{\mathrm{M}}, \boldsymbol{\Psi}^{\delta}\}$ [71].

Figure 2.5 shows the detailed flowchart of the modular Bayesian approach, which includes the following major distinctions with full Bayesian approach:

1. Module 1 replaces the computer code $\mathbf{y}^{\mathrm{M}}$ with a GP emulator, whose hyperparameters $\boldsymbol{\Psi}^{\mathrm{M}}$ is estimated based on the computational results at the training sites $\mathbf{y}^{\mathrm{M}}\left(\mathbf{x}^{\mathrm{IUQ}}, \boldsymbol{\Theta}\right)$. The evaluation process of $\boldsymbol{\Psi}^{\mathrm{M}}$ is straightforward using the methods described in Chapter 4, such as MLE.

2. Module 2 fits a second GP emulator to the model discrepancy term $\delta(\mathbf{x})$. The hyperparameters $\boldsymbol{\Psi}^{\delta}$ are estimated given the GP emulator from module 1, the measurement data $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}})$ and the prior of the calibration parameters $\boldsymbol{\theta}^{\mathrm{Prior}}$. Starting from the "model



Computer model $\mathbf{y}^{M}(\mathbf{x},\boldsymbol{\theta})$
$\boldsymbol{\theta}^{Prior}$
Experimental data $\mathbf{y}^{E}(\mathbf{x})$
Module 1: GP for computer model
$\mathbf{y}^{M}(\mathbf{x},\boldsymbol{\theta}) \sim GP\left\{\left(\mathbf{f}^{M}\right)^{T}\boldsymbol{\beta}^{M}, \sigma_{M}^{2}\Re^{M}\langle(\cdot,\cdot),(\cdot,\cdot)\rangle\right\}$
Module 2: GP for model discrepancy
$\delta(\mathbf{x}) \sim GP\left\{\left(\mathbf{f}^{\delta}\right)^{T}\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}\Re^{\delta}(\cdot,\cdot)\right\}$
Test source allocation
$(\boldsymbol{\Psi}^{M})^{Estimation}$
Model updating equation
$\mathbf{y}^{E}(\mathbf{x}^{IUQ}) = \mathbf{y}^{M}(\mathbf{x}^{IUQ},\boldsymbol{\theta}^{Prior}) + \delta(\mathbf{x}^{IUQ}) + \boldsymbol{\varepsilon}$
Inverse UQ data $\mathbf{y}^{E}(\mathbf{x}^{IUQ})$
$(\boldsymbol{\Psi}^{\delta})^{Estimation}$
Module 3: Posterior
$\boldsymbol{\theta}^{Posterior} = P\left(\boldsymbol{\theta}|\mathbf{y}^{E}(\mathbf{x}^{IUQ}), \left(\boldsymbol{\Psi}^{M}\right)^{Estimation}, \left(\boldsymbol{\Psi}^{\delta}\right)^{Estimation}\right)$
Forward uncertainty propagation
$\mathbf{y}^{R}(\mathbf{x}^{VAL}) = \mathbf{y}^{M}(\mathbf{x}^{VAL},\boldsymbol{\theta}^{Posterior}) + \delta(\mathbf{x}^{VAL})$
Validation metric
Validation
Validation data $\mathbf{y}^{E}(\mathbf{x}^{VAL})$
Prediction
$\mathbf{y}^{R}(\mathbf{x}^{PRED}) = \mathbf{y}^{M}(\mathbf{x}^{PRED},\boldsymbol{\theta}^{Posterior}) + \delta(\mathbf{x}^{PRED})$

Fig. 2.5 Workflow of the modular Bayesian approach.

updating equation", one forms a likelihood function for $\mathbf{y}^{E}\left(\mathbf{x}^{IUQ}\right)$ based on module 1 and $\boldsymbol{\theta}^{Prior}$ and then evaluates the MLEs of $\boldsymbol{\Psi}^{\delta}$. Closed forms of the likelihood function is possible using Gaussian correlation kernel, constant basis functions and normal [71] or uniform [4] priors for $\boldsymbol{\theta}^{Prior}$.

3. Once module 1 and module 2 become known, module 3 achieves the posterior $\boldsymbol{\theta}^{Posterior}$ from the likelihood function through MCMC sampling. Note that unlike the full Bayesian approach in which marginalization is required, here $\boldsymbol{\theta}^{Posterior}$ is conditioned on the estimations of the GP hyperparameters $\{\boldsymbol{\Psi}^{M},\boldsymbol{\Psi}^{\delta}\}$. However, this is also a limitation of the modular Bayesian approach, because the uncertainties in $\{\boldsymbol{\Psi}^{M},\boldsymbol{\Psi}^{\delta}\}$ are not considered as they have been fixed at plausible estimates. This is why such a method is only "empirical or partial" Bayesian [10], as opposed to "full" Bayesian.

Note that the definition of modules is not unanimous in literature. For example, [80] chose the (1) computer model $\mathbf{y}^{M}$, (2) the model discrepancy δ and (3) the field data $\mathbf{y}^{E}$ as three modules, while in [4], the (1) computer model $\mathbf{y}^{M}$, (2) model discrepancy δ, (3) posterior $\boldsymbol{\theta}^{Posterior}$ and (4) prediction of $\mathbf{y}^{E}$ and δ were treated as four modules. In the current work we use definition similar to [4] as shown in Figure 2.5. However, we do not treat evaluation of $\mathbf{y}^{\mathrm{E}}$ and δ in the prediction domain as a fourth module because extrapolated prediction is not a part of inverse UQ.

2.2.3 Comparison of full vs modular Bayesian approaches

Full Bayesian analysis is theoretically superior, but computationally intractable. There have been a lot of research using the modular Bayesian approach in literature [4] [10] [13] [57] [80] [144], while the work using full Bayesian is quite limited [61] [62] [63]. Full Bayesian requires that the user have reasonably good priors for the GP hyperparameters $\{\mathbf{\Psi}^{\mathrm{M}}, \mathbf{\Psi}^{\delta}\}$. This is very difficult in practice especially for the model discrepancy term. The joint posterior for all the unknowns shown in Equation 2.11 can be extremely complicated and have very high dimension, posing challenges for MCMC sampling and subsequent marginalization.

Full Bayesian analysis takes into account the uncertainties in the hyperparameters. However, for the calibration-validation-prediction process, the uncertainties in $\boldsymbol{\theta}$ and $\mathbf{\Psi}^{\delta}$ tend to overwhelm the uncertainties in computer model approximations (i.e. $\mathbf{\Psi}^{\mathrm{M}}$). It was found that using MLE plug-in typically yields much the same answers as the full Bayesian. Therefore, researchers recommended using the partial/empirical Bayesian rather than full Bayesian [10].

Liu et al. [80] also mentioned some other advantages of modularization. For example, separating good modules from suspect modules can avoid information "contamination". Suspect modules are modules that are unknown or improperly specified (especially when there is no better choice). Moreover, modularization reduces the computational complexity by reasonable simplification. It also makes scientific understanding and development more convenient. Mixing and convergence of MCMC sampling can be improved by modularization. Finally, modularization also helps alleviate the issue of confounding or lack of identifiability.

2.2.4 Discussions on the modeling of model discrepancy term

In Section 2.1, we have briefly talked about the model discrepancy term $\delta(\mathbf{x})$. Model discrepancy accounts for inadequacies that are caused by missing or insufficient physics and numerical approximations built into the simulation model, which leads to systematic differences between the simulation model and the reality [61]. Ignoring model discrepancy during inverse UQ will cause over-fitting and subsequent prediction errors. However, due to the inherent difficulty in the mathematical description of the model discrepancy or limited amount of data, in many previous work on Bayesian calibration it is simply ignored [82] [91] [89] [87] [129] [139] [150] [156] [155]. Some representative work that consider model discrepancy are [4] [61] [62] [63] [71] [144].

The primary challenge in dealing with model discrepancy is that there are no direct observations from the discrepancy. Another difficulty is that model discrepancy is almost always seriously "confounded" with other model unknowns such as the calibration parameters [80]. All of the previous research follows the full or modular Bayesian approaches described above to provide mathematical descriptions of the model discrepancy term. Such a choice has been proven successful in many applications as shown in the aforementioned publications. However, there is an important issue associated with such a method: **extrapolation** of the model discrepancy to validation and prediction domains.

As shown in Figure 2.4 and 2.5, after learning about $\boldsymbol{\theta}^{\text{Posterior}}$, the realities in the validation and prediction domain are described as:

$$\mathbf{y}^{\text{R}}\left(\mathbf{x}^{\text{VAL}}\right) = \mathbf{y}^{\text{M}}\left(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^{\text{Posterior}}\right) + \delta\left(\mathbf{x}^{\text{VAL}}\right) \tag{2.12}$$

$$\mathbf{y}^{\text{R}}\left(\mathbf{x}^{\text{PRED}}\right) = \mathbf{y}^{\text{M}}\left(\mathbf{x}^{\text{PRED}}, \boldsymbol{\theta}^{\text{Posterior}}\right) + \delta\left(\mathbf{x}^{\text{PRED}}\right) \tag{2.13}$$

where $\mathbf{x}^{\text{VAL}}$ and $\mathbf{x}^{\text{PRED}}$ stand for the designs variables in the validation and prediction domain respectively. The first term on the right $\mathbf{y}^{\text{M}}$ can be either the computer model itself or the GP emulator, while the second term δ is always the GP emulator. Because the information acquired about the model discrepancy hyperparameters $(\boldsymbol{\Psi}^{\delta})^{\text{Posterior}}$ or $(\boldsymbol{\Psi}^{\delta})^{\text{Estimation}}$ is based on model simulation $\mathbf{y}^{\text{M}}(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\Theta})$ and measurement data $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{IUQ}})$ on the inverse UQ domain, Equation 2.12 and 2.13 involve extrapolation of such information to the validation and prediction domain.

Inverse UQ with both full and modular Bayesian are fully data-driven. Therefore, these methods should be used with great caution. Extrapolation outside the range of the inverse UQ domain is questionable. As discussed in [61], the quality of such extrapolation largely depends on the reliability of the model discrepancy term. What we have learnt about the model discrepancy at the inverse UQ domain may not be applicable to the validation and prediction domain. See [63] for an example in which the model discrepancy is large in magnitude, but the posterior distribution is similar with that obtained when the model discrepancy is zero.

Besides the reliability of the model discrepancy term, extrapolation using GP emulator is inherently dangerous. GP emulator usually has large mean prediction errors and significant variance outside of the training domain. Even though the full and modular Bayesian methodologies outlined above have been accepted for a long time, we recommend making some improvement of the description of the model discrepancy term to avoid extrapolation.

2.3 Improved Modular Bayesian Approach

An improved modular Bayesian approach is developed and described in this section. It is much more straightforward to understand and apply, and it is capable of avoiding the issue of extrapolation of the GP emulators. We also use a GP emulator to represent the model discrepancy term $\delta(\mathbf{x}) \sim \mathcal{GP}\left\{\left(\mathbf{f}^{\delta}\right)^{\top}\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}\mathcal{R}^{\delta}\right\}$. To solve for the GP hyperparameters $\boldsymbol{\Psi}^{\delta} = \left\{\boldsymbol{\beta}^{\delta}, \sigma_{\delta}^{2}, \boldsymbol{\theta}^{\delta}, \mathbf{p}^{\delta}\right\}$ we need training data whose input is $\mathbf{x}$ and output is the computer code simulation error (recall that model discrepancy is also called model error/bias/uncertainty/inadequacy). Because there is no direct observations of $\delta(\mathbf{x})$, we need substitutes of such "observation data for simulation error". Three intuitively natural modularization schemes were compared in [80] to fix $\boldsymbol{\Psi}^{\delta}$.

1. Modular approach 1: Treat the differences between experimental observations and model simulations (run at prior means or nominal values of calibration parameters, but using the same design variables with measurement data) as the realizations of model discrepancy. Then $\boldsymbol{\Psi}^{\delta}$ can be estimated with methods like MLE and fixed.

2. Modular approach 2: Sample the calibration parameters from their prior distributions. Then sample $\boldsymbol{\Psi}^{\delta}$ from their posteriors, which is generated conditioning on the samples of $\boldsymbol{\theta}^{\text{Prior}}$. The posterior sample mean will be used as the fixed values of $\boldsymbol{\Psi}^{\delta}$.

3. Modular approach 3: Initially assume the model is perfect (model discrepancy is zero). Then solve for $\boldsymbol{\theta}^{\text{Posterior}}$ based on this assumption. The resulting posterior will be used as a "new" prior and proceed with modular approach 1.

Modular approach 1 was demonstrated to have the best performance for the simple test problem in [80]. Actually such treatment has been used in other related research, for example design-driven validation [20] and Bayesian validation [141]. In the examples of [141], it was shown that the model discrepancy posterior using the modular Bayesian approach is same as that obtained by fitting a single GP to $\mathbf{y}^{\text{E}} - \mathbf{y}^{\text{M}}$. Whether this conclusion applies to more complicated real problem is unknown.

The revised modular Bayesian approach takes the modular approach 1 and its main components are shown in Figure 2.6, which consists of the following steps:

1. Step 1: separate the measurement data for inverse UQ and validation:
   This step is shown as black arrows in Figure 2.6. Instead of random selection, we use a carefully designed an sequential algorithm for TSA, which will be discussed in Chapter 7. For now, we assume that the given tests have already been separated.

2. Step 2: mathematical description of the model discrepancy term $\delta(\mathbf{x})$:
   This step is shown as green arrows in Figure 2.6. The computer code $\mathbf{y}^{\text{M}}(\mathbf{x}, \boldsymbol{\theta})$ is first

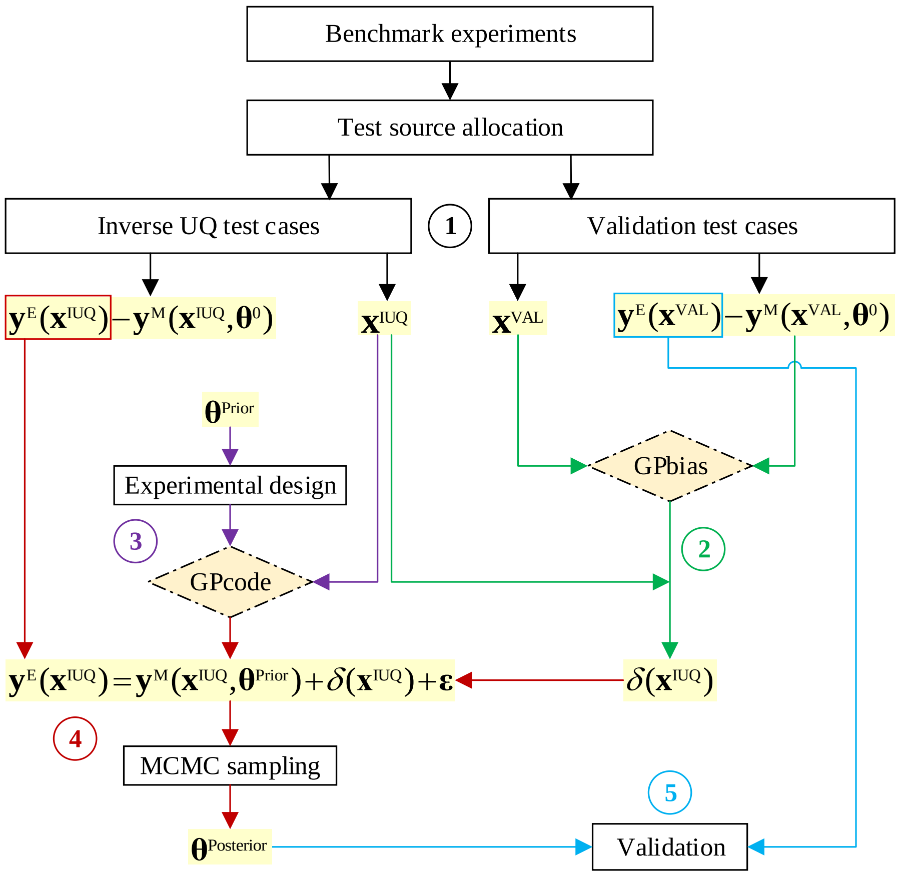

Benchmark experiments
Test source allocation
Inverse UQ test cases
1
Validation test cases
$\mathbf{y}^E(\mathbf{x}^{IUQ}) - \mathbf{y}^M(\mathbf{x}^{IUQ}, \boldsymbol{\theta}^0)$
$\mathbf{x}^{IUQ}$
$\mathbf{x}^{VAL}$
$\mathbf{y}^E(\mathbf{x}^{VAL}) - \mathbf{y}^M(\mathbf{x}^{VAL}, \boldsymbol{\theta}^0)$
$\boldsymbol{\theta}^{Prior}$
Experimental design
GPbias
3
2
GPcode
$\mathbf{y}^E(\mathbf{x}^{IUQ}) = \mathbf{y}^M(\mathbf{x}^{IUQ}, \boldsymbol{\theta}^{Prior}) + \delta(\mathbf{x}^{IUQ}) + \boldsymbol{\varepsilon}$
$\delta(\mathbf{x}^{IUQ})$
4
MCMC sampling
5
$\boldsymbol{\theta}^{Posterior}$
Validation

Fig. 2.6 Flowchart of the improved modular Bayesian approach

executed at the input settings of all the tests $\mathbf{x}^{\text{IUQ}} \cup \mathbf{x}^{\text{VAL}}$, with the calibration parameters fixed at nominal values or prior mean values $\boldsymbol{\theta}^0$ which can be viewed as our current best knowledge of $\boldsymbol{\theta}$. The resulting simulations are denoted as $\mathbf{y}^{\text{M}}(\mathbf{x}^{\text{test}}, \boldsymbol{\theta}^0)$. Then $\mathbf{x}^{\text{VAL}}$ and $\left\{\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^0)\right\}$ are used as training inputs and output respectively to fit a GP emulator called `GPbias`. Evaluating `GPbias` at $\mathbf{x}^{\text{IUQ}}$ results in an estimation of the model discrepancy term $\delta\left(\mathbf{x}^{\text{IUQ}}\right)$, which will enter the likelihood function during MCMC sampling.

3. Step 3: fit a GP emulator for the computer code:
   This step is shown as purple arrows in Figure 2.6. Another GP emulator called `GPcode` is fitted to replace the computer code during MCMC sampling. `GPcode` uses $(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^{\text{Prior}})$ as training inputs and $\mathbf{y}^{\text{M}}(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^{\text{Prior}})$ as training outputs. `GPcode` needs to be built with an experimental design of $\boldsymbol{\theta}$ following the distributions of $\boldsymbol{\theta}^{\text{Prior}}$.

4. Step 4: MCMC sampling:
   This step is shown as red arrows in Figure 2.6. `GPcode`, $\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)$ and $\mathbf{y}^{\mathrm{E}}\left(\mathbf{x}^{\mathrm{IUQ}}\right)$ enters the posterior PDF which is then explored with MCMC sampling to achieve $\boldsymbol{\theta}^{\mathrm{Posterior}}$.

5. Step 5: validation of $\boldsymbol{\theta}^{\mathrm{Posterior}}$:
   This step is shown as light blue arrows in Figure 2.6. Computer code simulations based on $\boldsymbol{\theta}^{\mathrm{Posterior}}$ and $\mathbf{y}^{\mathrm{E}}\left(\mathbf{x}^{\mathrm{VAL}}\right)$ are compared for validation with certain validation metrics.

Note that in the validation domain (and future prediction domain as well), we use $\mathbf{y}^{\mathrm{R}}\left(\mathbf{x}^{\mathrm{VAL}}\right) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})$ instead of $\mathbf{y}^{\mathrm{R}}(\mathbf{x}^{\mathrm{VAL}}) = \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}}) + \delta(\mathbf{x}^{\mathrm{VAL}})$ to avoid extrapolation. Because of the specially designed TSA process in step 1 (see details Chapter 7) and treatment of model discrepancy in step 2, $\boldsymbol{\theta}^{\mathrm{Posterior}}$ is informed by the experimental data in the validation tests so that the model discrepancy term $\delta(\mathbf{x}^{\mathrm{VAL}})$ is no longer needed to correct the model simulation $\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})$. Such treatment of model discrepancy term is novel and in Chapter 7, we will demonstrate that it is capable of avoid over-fitting, even when only partial of the QoIs are used for inverse UQ.

2.4 Markov Chain Monte Carlo Sampling

2.4.1 General Introduction

MCMC [45] methods are commonly used to numerically approximate integrals of the following form:

$$\mathcal{I}(f) = \int f(x)\pi(x)dx \tag{2.14}$$

where $\pi(x)$ is the target probability density function, and the objective is to produce a set of random samples $(x_i)_{i=1}^N$ from the target distribution π and approximate the integral of $\mathcal{I}(f)$ by $\frac{1}{N}\sum_{i=1}^N f(x_i)$. $(x_i)_{i=1}^N$ is called a Markov chain, with π defined as the unique invariant distribution.

MCMC is widely used to sample from complicated distributions without explicitly knowing the normalizing constant. The most popular algorithm is the Metropolis-Hastings (MH) algorithm [45], which defines a family of possible transitions from one Markov chain state to the next from a proposal distribution (e.g. Gaussian). Algorithm 0 shows a popular choice of MH algorithm in which the **Symmetric Random Walk Metropolis algorithm (SRWM)** is used to produce transitions [3].

Algorithm 0 Metropolis-Hastings (MH) algorithm

```
1: Initialize $x_0$
2: Choose appropriate proposal distribution $g(x^*|x)$
3: for iteration $i+1$, $i \geq 0$, given $x_i$ do
4:     Proposal a sample $\xi \sim g(\xi|x_i)$
5:     Calculate the acceptance probability $\alpha = \min\left[1, \frac{\pi(\xi)\cdot g(x_i|\xi)}{\pi(x_i)\cdot g(\xi|x_i)}\right]$
6:     Sample $\eta \sim \text{uniform}(0,1)$
7:     if $\alpha \geq \eta$ then
8:         Accept the proposed sample, $x_{i+1} = \xi$
9:     else Reject the proposed sample, stay at the current location, $x_{i+1} = x_i$
10:    end if
11: end for
```

2.4.2 Adaptive MCMC Sampling

Although the symmetric random-walk MH algorithm is simple in design and widely applicable (even in high-dimension problems), the convergence is usually very slow and special attention is required to adjust the acceptance rate. Adaptive methods are useful in tuning critical parameters that are necessary for efficient mixing. One approach for an adaptive method is to optimally adjust the variance of transition probability, however without care the adaptive process will lose its ergodic properties (consistency of estimates and convergence to the target distribution). Therefore, rules must be defined to provide acceptable approximate sampling methods of optimizing the transition probability to ensure that ergodicity is maintained asymptotically.

Andrieu and Thoms [3] outlined several algorithms that incrementally implement adaptive MCMC techniques, and in the current study three adaptive algorithms have been implemented to examine their practical capabilities.

The first algorithm, called the "**Adaptive Metropolis (AM)**" algorithm, uses the classical multivariate Gaussian as the proposal distribution and recursively updates the distribution covariance to converge to the optimal choice of the true covariance of the target distribution. In this algorithm, the covariance matrix of the proposal distribution is scaled by λ. It is shown [45] that the "optimal" covariance matrix for the normal symmetric random walk MH algorithm is $\frac{2.38^2}{N_x} \cdot \Sigma_\pi$ where Σ_π is the true covariance matrix of the target distribution π and N_x is the dimension of the input space. This value is later used as the "optimal scaling factor" in AM algorithm by [55]. In this way, the true covariance matrix Σ_π is learned "on-the-fly". Note that in Algorithm 1 the value of the scaling factor is not updated.

The second algorithm ("**Rao-Blackwellized AM**") [3] takes the Rao-Blackwell approach and adjusts the estimator to update the covariance matrix depending on weighted averages using the current acceptance probability. The adjusted recursions for the mean and variance are

Algorithm 1 Adaptive Metropolis (AM) algorithm

```
1: Initialize $x_0$, $\mu_0$ and $\Sigma_0$
2: for iteration $i+1$, $i \geq 0$, given $x_i$, $\mu_i$ and $\Sigma_i$ do
3:     Proposal a sample $\xi \sim \mathcal{N}(x_i, \lambda\Sigma_i)$
4:     Accept the proposed sample with probability $\alpha(x_i, \xi)$, $x_{i+1} = \xi$; otherwise keep the
   current sample, $x_{i+1} = x_i$
5:     Update:
           $\mu_{i+1} = \mu_i + \gamma_{i+1}(x_{i+1} - \mu_i)$
           $\Sigma_{i+1} = \Sigma_i + \gamma_{i+1}\left[(x_{i+1} - \mu_i)(x_{i+1} - \mu_i)^\intercal - \Sigma_i\right]$
6: end for
```

explicitly defined as:

$$\mu_{i+1} = \mu_i + \gamma_{i+1}\left[\alpha(x_i, \xi) \cdot (\xi - \mu_i) + (1 - \alpha(x_i, \xi)) \cdot (x_i - \mu_i)\right] \tag{2.15}$$

$$\begin{aligned}\Sigma_{i+1} = \Sigma_i + \gamma_{i+1}[\alpha(x_i, \xi) \cdot (\xi - \mu_i)(\xi - \mu_i)^\intercal + \\ (1 - \alpha(x_i, \xi)) \cdot (x_{i+1} - \mu_i)(x_{i+1} - \mu_i)^\intercal - \Sigma_i]\end{aligned} \tag{2.16}$$

We define the short notation:

$$\overline{x_{i+1}} = \alpha(x_i, \xi) \cdot \xi + (1 - \alpha(x_i, \xi)) \cdot x_i \tag{2.17}$$

$$\begin{aligned}\overline{(x_{i+1} - \mu_i)(x_{i+1} - \mu_i)^\intercal} = \alpha(x_i, \xi) \cdot (\xi - \mu_i)(\xi - \mu_i)^\intercal + \\ (1 - \alpha(x_i, \xi)) \cdot (x_{i+1} - \mu_i)(x_{i+1} - \mu_i)^\intercal\end{aligned} \tag{2.18}$$

The steps for Rao-Blackwellized AM is shown in Algorithm 2.

Algorithm 2 Rao-Blackwellized AM algorithm

```
1: Initialize $x_0$, $\mu_0$ and $\Sigma_0$
2: for iteration $i+1$, $i \geq 0$, given $x_i$, $\mu_i$ and $\Sigma_i$ do
3:     Proposal a sample $\xi \sim \mathcal{N}(x_i, \lambda\Sigma_i)$
4:     Accept the proposed sample with probability $\alpha(x_i, \xi)$, $x_{i+1} = \xi$; otherwise keep the
   current sample, $x_{i+1} = x_i$
5:     Update:
           $\mu_{i+1} = \mu_i + \gamma_{i+1}(\overline{x_{i+1}} - \mu_i)$
           $\Sigma_{i+1} = \Sigma_i + \gamma_{i+1}\left[\overline{(x_{i+1} - \mu_i)(x_{i+1} - \mu_i)^\intercal} - \Sigma_i\right]$
6: end for
```

The third algorithm ("**AM with global adaptive scaling**") [3] attempts to adjust the scaling of proposal covariance matrix to adjust the sampling to a target acceptance probability, rather than using a pre-set constant for the scaling factor . Specifically, the log of the scaling parameter is adjusted recursively to the predefined "optimal acceptance rate" α^{opt} (e.g. 0.234 is recommended) [55]. This allows the algorithm to control its rate of exploration by adjusting the scale if the acceptance rate is lower or higher than the optimum acceptance rate. The advantage of this algorithm is that one might expect a more rapid exploration of the target distribution following a poor initialization.

Then Algorithm 3 consists of the following steps:

Algorithm 3 AM algorithm with global adaptive scaling

```
1: Initialize $x_0$, $\mu_0$ and $\Sigma_0$
2: for iteration $i+1$, $i \geq 0$, given $x_i$, $\mu_i$ and $\Sigma_i$ do
3:     Proposal a sample $\xi \sim \mathcal{N}(x_i, \lambda\Sigma_i)$
4:     Accept the proposed sample with probability $\alpha(x_i,\xi)$, $x_{i+1} = \xi$; otherwise keep the
       current sample, $x_{i+1} = x_i$
5:     Update:
           $\log(\lambda_{i+1}) = \log(\lambda_i) + \gamma_{i+1}\left(\alpha(x_i,\xi) - \alpha^{\text{opt}}\right)$
           $\mu_{i+1} = \mu_i + \gamma_{i+1}(\overline{x_{i+1}} - \mu_i)$
           $\Sigma_{i+1} = \Sigma_i + \gamma_{i+1}\left[\overline{(x_{i+1}-\mu_i)(x_{i+1}-\mu_i)^\intercal} - \Sigma_i\right]$
6: end for
```

There are many other adaptive methods available, such as "**Component-wise AM**" [3]. Component-wise adaption addresses the issue that adaptive scaling may not be efficient in all directions simultaneously, and strategizes to use "timid" moves to initiate sampling. However, in our experience these adaptive methods will not capture the correlation between different input parameters well enough, since the chains for highly correlated parameters usually do not mix well. Another issue is that most component-wise adaptive algorithms require posterior evaluation for every dimension when a new sample is proposed. In this case, the algorithm will be at least times more expensive than Algorithms 1, 2 and 3 listed above. Therefore, component-wise adaptive method is not considered appropriate for this study.

2.5 Numerical Studies

In this thesis, we will use four examples to test the developed approach for inverse UQ.

1. Chapter 5: a simplified nuclear reactor simulation model, which is the Point Reactor Kinetics Equation (PRKE) coupled with lumped parameter TH feedback model based on synthetic experimental data, with Polynomial Chaos Expansion (PCE) surrogate model.

2. Chapter 6: best-estimate system TH code TRACE physical model parameters based on OECD/NRC BWR Full-size Fine-Mesh Bundle Tests (BFBT) benchmark steady-state void fraction data, with Sparse Grid Stochastic Collocation (SGSC) surrogate model.
3. Chapter 7: the same problem with Chapter 6 but with GP emulator.
4. Chapter 8: fuel performance code BISON Fission Gas Release (FGR) model based on Risø-AN3 on-line time-dependent measurement data, with GP emulator.

Note that because of the data available, only the application in Chapter 7 fully follow the improved modular Bayesian approach developed in Section 2.3 and considers model discrepancy. Application in Chapter 5 and 6 aim to demonstrate the power of stochastic spectral surrogate models (PCE and SC). The example in Chapter 8 deals with the case when time series data is available and good agreement cannot be achieved with any combinations of calibration parameters because the shapes of simulation and data have drastic difference.

Chapter 3

STOCHASTIC SPECTRAL METHODS

In this chapter, details of the stochastic spectral techniques will be presented. We introduce the generalized Polynomial Chaos Expansion (PCE) and Stochastic Collocation (SC), as well as the methods to solve for generalized PCE and Sparse Grid Stochastic Collocation (SC).

3.1 Overview of Stochastic Spectral Techniques and Their Applications

While an extraordinary progress was made during the past decade in developing advanced concepts, methods and tools for uncertainty and sensitivity analysis, there remains a significant gap in bringing these advances to nuclear engineering practice, particularly in the areas of nuclear TH, fuel performance and multi-physics simulation. In recent years a growing emphasis has been given to stochastic spectral methods based on generalized Polynomial Chaos Expansion (PCE) and Stochastic Collocation (SC) methods due to their relatively simple application and excellent accuracy, efficiency and convergence. Stochastic spectral techniques approximate stochastic processes involved in the mathematical model by means of a spectral expansion in the random space. Figure 3.1 shows the hierarchy of stochastic spectral methods.

3.1.1 Generalized Polynomials Chaos Expansion

PCE is a method that expands the model outputs with respect to orthogonal polynomials in the random model inputs and then projects the expansion onto the space spanned by the same orthogonal polynomials [78] [161]. This stochastic projection provides a compact and convenient representation of the model output variability with respect to the random inputs. The idea for this spectral representation was first introduced by Wiener to represent Gaussian

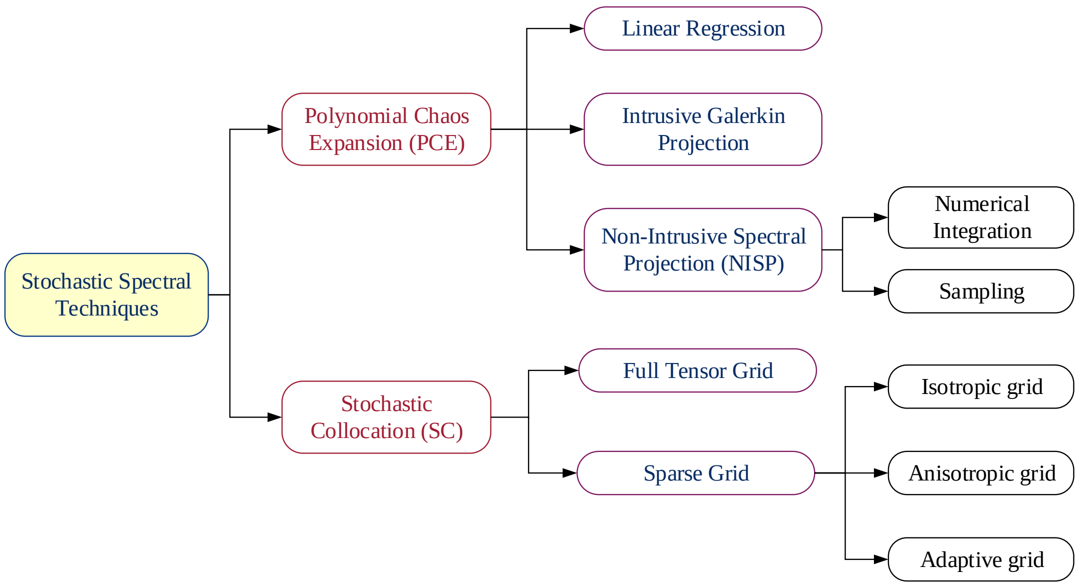

Stochastic Spectral Techniques
Polynomial Chaos Expansion (PCE)
Stochastic Collocation (SC)
Linear Regression
Intrusive Galerkin Projection
Non-Intrusive Spectral Projection (NISP)
Numerical Integration
Sampling
Full Tensor Grid
Sparse Grid
Isotropic grid
Anisotropic grid
Adaptive grid

Fig. 3.1 Hierarchy of stochastic spectral methods

processes with Hermite polynomials [143]. It was later extended by Xiu and Karniadakis [164] so that other types of stochastic processes could also be addressed. The Wiener-Askey scheme [164] consists of a family of orthogonal polynomials which can be used to represent different family of stochastic processes. For example, Hermite polynomials are best suited to represent Gaussian random variables, Legendre polynomials have better performance for uniform distributions. The generalization of Wiener's original "Homogeneous Chaos" theory [143] results in the generalized PCE technique that is widely used today.

Ghanem and co-workers first applied generalized PCE for UQ of solid mechanics problems [48] [49]. Later, generalized PC found application in model uncertainty in diffusion [163] and fluid flow [97] [165]. Error bounds and convergence studies [29] [34] have shown that these methods exhibit fast convergence rates with increasing orders of expansion. The convergence studies assume that the solution is sufficiently smooth in the random space.

Applying stochastic spectral techniques means determining the spectral expansion coefficients. These techniques can be categorized as intrusive and non-intrusive [30] [31] [32], depending on whether it requires modification of the original model (e.g. source code) for its application. The PCE method for the forward uncertainty propagation involves the substitution of uncertain variables in the governing equations with their polynomial expansions. In general, an intrusive approach will calculate the unknown polynomial coefficients by projecting the resulting equations onto a basis functions (orthogonal polynomials) for different modes (expansion coefficients). With intrusive approaches (mainly referred to as Galerkin projection), a new mathematical problem is defined after the Galerkin projection, where the solution is the set of expansion coefficients. A new solver is required and this may be prohibitive for many complex computational problems where modification of an existing code is too difficult and time consuming. Depending on the dimension of the original problem uncertain inputs and the order of PCE, the new mathematical system can be significantly larger than the original one.

Generalized PCE also suffers from the so-called **Curse of dimensionality** which means that the number of simulations required increases exponentially with the dimensions of computer models. This makes generalized PCE most applicable for low-to-moderate-dimensional problems. Despite these difficulties, PCE based on Galerkin projection approach exhibit several substantial advantages over MC sampling and perturbation methods. Spectral convergence rates are achieved for most conditions [164]. Furthermore, in cases where non-linearities and large uncertainties in random variables are a problem for perturbation methods, PCE can deal with these problems much easier.

On the other hand, non-intrusive techniques use the original model as a "black box" and the calculation of PCE coefficients for the outputs is based on a set of model response evaluations. Non-intrusive methods may be categorized as either Non-Intrusive Spectral Projection (NISP) methods, or regression-based approach [31] [32]. Numerical integration methods may be further categorized as either tensor product quadrature or sparse grid Smolyak cubature, which can be sub-divided further as either isotropic or anisotropic.

3.1.2 Stochastic Collocation

SC [5] [160] [162] is another stochastic expansion technique that is closely related to PCE. Whereas generalized PCE estimates coefficients for known orthogonal polynomial basis functions, SC forms Lagrange interpolation functions for known coefficients [30] [31]. The SC expansion is formed as a sum of a set of multidimensional Lagrange interpolation polynomials, one polynomial per collocation point. Since these polynomials have the feature of being equal to 1 at their particular collocation point and 0 at all other points, the coefficients of the expansion are just the response values at each of the collocation points.

The implementation of SC methods is straight-forward as it only requires solution of the corresponding deterministic problem at each interpolation point, similar to MC sampling method at each sampling point. Such property makes SC methods good alternative to MC sampling and intrusive Galerkin-based PCE. The core issue is the construction of a set of interpolation points, and it is non-trivial in multidimensional random space. For such cases, point sets based on tensor products of one-dimensional quadrature points are not suitable, as the number of points grows too fast with increasing dimensions. This difficulty is alleviated by the application of sparse grid interpolation method based on the Smolyak algorithm [8] [46] [47] [75] [127]. Further improvements include the application of isotropic, anisotropic and adaptive sparse grids [22] [23] [40] [101] [135], which can reduce the number of interpolation nodes by several orders of magnitude for high-dimensional problems.

In essence, stochastic spectral methods approximate a model output as a polynomial function of random input parameters. Such a function of the output constitutes a meta-model of the original problem and can be used to describe stochastic nature of the output in terms of its mean, variance, covariance, distribution, etc..

3.2 Polynomial Chaos Expansion

3.2.1 Properties of Orthogonal Polynomials

Suppose we have a system of polynomials $\{Q_n(x), n \in \mathbb{N}\}$ where $Q_n(x)$ is a polynomial of exact degree n and $\mathbb{N} = \{0, 1, 2, \dots\}$. This system of polynomials is called *orthogonal* with respect to some real positive measure $\phi(x)$ if the following orthogonality relation is satisfied:

$$\int_S Q_n(x) Q_m(x) d\phi(x) = h_n^2 \delta_{mn}, \quad m, n \in \mathbb{N} \tag{3.1}$$

where S is the support of the measure $\phi(x)$ and h_n's are non-zero constants. The system is called orthonormal if $h_n = 1$. δ_{mn} is the Kronecker delta. The measure $\phi(x)$ usually has a density of $w(x)$ for continuous case or weights w_i at points x_i in the discrete case.

Equation 3.1 can be written as:

$$\int_S Q_n(x) Q_m(x) w(x) dx = h_n^2 \delta_{mn}, \quad m, n \in \mathbb{N} \tag{3.2}$$

in the continuous case, or

$$\sum_{i=0}^{M} Q_n(x_i) Q_m(x_i) w_i = h_n^2 \delta_{mn}, \quad m, n \in \mathbb{N} \tag{3.3}$$

for discrete case.

The density function $w(x)$ or weights w_i are also commonly referred to as the *weighting function*. In the next section, it will be shown that the weighting functions for some orthogonal polynomials are similar to certain probability functions. For example, the weighting function for the Hermite polynomials differs with the PDF formula of the Gaussian distribution by only a constant. This fact plays an important role in representing stochastic processes with orthogonal polynomials.

3.2.2 The Homogeneous Chaos

The original polynomial chaos, also called the *homogeneous chaos*, was first introduced by Wiener [143]. Homogeneous chaos provides a way for expanding second-order random processes in terms of Hermite polynomials. Second-order random processes are processes with finite variance, which applies to most physical processes. Define a probability space as $(\Omega, \mathcal{F}, \mathcal{P})$, where Ω is the sample space, $\mathcal{F}$ is the σ-algebra over Ω and $\mathcal{P}$ is a probability measure defined on $\mathcal{F}$. For a set of orthonormal standard Gaussian random variables $\{\xi_i(\omega)\}_{i=1}^{\infty}$ on $\mathcal{F}$, any second-order random process $X(\omega) : \Omega \to \mathbb{R}$ can be represented as:

$$X(\omega) = a_0\Gamma_0 + \sum_{i_1=1}^{\infty} a_{i_1}\Gamma_1(\xi_{i_1}(\omega)) + \sum_{i_1=1}^{\infty}\sum_{i_2=1}^{i_1} a_{i_1 i_2}\Gamma_2(\xi_{i_1}(\omega), \xi_{i_2}(\omega)) \\ + \sum_{i_1=1}^{\infty}\sum_{i_2=1}^{i_1}\sum_{i_3=1}^{i_2} a_{i_1 i_2 i_3}\Gamma_3(\xi_{i_1}(\omega), \xi_{i_2}(\omega), \xi_{i_3}(\omega)) + \cdots \quad (3.4)$$

In Equation 3.4, $\Gamma_n(\xi_{i_1}, \xi_{i_2}, \ldots, \xi_{i_n})$ denotes the Hermite polynomials of order n in terms of $\boldsymbol{\xi} = [\xi_{i_1}, \xi_{i_2}, \xi_{i_3}, \ldots, \xi_{i_n}]^\top$. The constants $a_{i_1 i_2 \ldots i_n}$ are the expansion coefficients, also called **modes**. Equation 3.4 is the discrete version of the original Wiener homogeneous chaos, where the continuous integrals are replaced by summations. The general expression of a multivariate Hermite polynomial $\Gamma(\boldsymbol{\xi})$ of order n is given by:

$$\Gamma_n(\xi_{i_1}, \xi_{i_2}, \ldots, \xi_{i_n}) = e^{\frac{1}{2}\boldsymbol{\xi}^\top\boldsymbol{\xi}} \cdot (-1)^n \cdot \frac{\partial^n}{\partial\xi_{i_1}\partial\xi_{i_2}\ldots\partial\xi_{i_n}}\left(e^{\frac{1}{2}\boldsymbol{\xi}^\top\boldsymbol{\xi}}\right) \quad (3.5)$$

The one-dimensional Hermite polynomials $\Gamma_n(\xi)$ are:

$$\begin{aligned} \Gamma_0(\xi) &= 1 \\ \Gamma_1(\xi) &= \xi \\ \Gamma_2(\xi) &= \xi^2 - 1 \\ \Gamma_3(\xi) &= \xi^3 - 3\xi \quad \ldots \end{aligned}$$

For notation convenience, the expansion shown in Equation 3.4 can be simplified by reformulating the equation from an order-based indexing to a term-based indexing:

$$X(\omega) = \sum_{i=0}^{\infty} x_i \Psi_i(\boldsymbol{\xi}) \quad (3.6)$$

There is a one-to-one correspondence between the expansion coefficients $a_{i_1 i_2 \ldots i_n}$ and x_i , as well as between expansion polynomials $\Gamma_n\left(\xi_{i_1}, \xi_{i_2}, \ldots, \xi_{i_n}\right)$ and Ψ_i. Each of the Ψ_i is a multi-dimensional Hermite polynomial and can be generated from univariate Hermite polynomials by taking tensor products. This new expansion in Equation 3.6 is simply a re-numbering with the polynomials of lower order counted first.

The Hermite polynomial chaos forms a complete orthogonal basis in the L_2 space of the Gaussian random variable:

$$\left\langle \Psi_i, \Psi_j \right\rangle = \left\langle \Psi_i^2 \right\rangle \delta_{ij} \tag{3.7}$$

where $\langle \cdot, \cdot \rangle$ denotes the ensemble average. This is the inner product in the Hilbert space of the Gaussian random variable $f(\boldsymbol{\xi})$ and $g(\boldsymbol{\xi})$:

$$\langle f(\boldsymbol{\xi}), g(\boldsymbol{\xi}) \rangle = \int f(\boldsymbol{\xi}) g(\boldsymbol{\xi}) \omega(\boldsymbol{\xi}) d\boldsymbol{\xi} \tag{3.8}$$

in which the weighting function $\omega(\boldsymbol{\xi})$ is:

$$\omega(\boldsymbol{\xi}) = \frac{1}{\sqrt{(2\pi)^n}} \cdot e^{\frac{1}{2}\boldsymbol{\xi}^\top \boldsymbol{\xi}} \tag{3.9}$$

where n is the dimension of $\boldsymbol{\xi}$. What distinguishes the Wiener–Hermite expansion from many other possible complete sets of orthogonal polynomial expansions is that the polynomials here are orthogonal with respect to the weighting function $\omega(\boldsymbol{\xi})$ which has the form of the multi-dimensional independent Gaussian probability distribution with unit variance. We will use the term Hermite chaos hereafter to denote the Wiener polynomial chaos.

3.2.3 The Generalized PCE

In order to deal with more general random distributions, the generalized PCE was developed. It was derived from the family of hypergeometric orthogonal polynomials known as the *Askey scheme* [164], for which the Hermite polynomials originally employed by Wiener are a subset. Table 3.1 presents the correspondence of different type of continuous Wiener-Askey polynomials to different type of probability distributions. The density and weighting functions differ by a constant due to the requirement that the integral of the PDF over the support range is one.

In practice, the infinite expansion is truncated at a finite number of random variables and a finite expansion order:

$$X(\boldsymbol{\xi}) = \sum_{i=0}^{P} x_i \Psi_i(\boldsymbol{\xi}) \tag{3.10}$$

Table 3.1 Correspondence of the various continuous Wiener-Askey polynomial chaos to different type of random distributions

Distribution	PDF	Orthogonal polynomial	Weight function	Support
Normal	$\frac{1}{2\pi}e^{-\frac{x^2}{2}}$	Hermite $He_n(x)$	$e^{-\frac{x^2}{2}}$	$(-\infty,\infty)$
Gamma	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$	Generalized Laguerre $L_n^\alpha(x)$	$x^\alpha e^{-x}$	$[0,\infty)$
Exponential	e^{-x}	Laguerre $L_n(x)$	e^{-x}	$[0,\infty)$
Beta	$\frac{(1-x)^\alpha(1+x)^\beta}{2^{\alpha+\beta+1}B(\alpha+1,\beta+1)}$	Jacobi $P_n^{\alpha,\beta}(x)$	$(1-x)^\alpha(1+x)^\beta$	$[-1,1]$
Uniform	$\frac{1}{2}$	Legendre $P_n(x)$	1	$[-1,1]$

Since each type of polynomial from the Askey scheme forms a complete basis in the Hilbert space determined by their corresponding support, we can expect each type of Askey-chaos to converge to any L_2 functional in the L_2 sense in the corresponding Hilbert functional space as a generalized result of Cameron–Martin theorem [18].

There are $(P+1)$ expansion terms, which depends on the dimension N of $\boldsymbol{\xi}$ and the highest order d of the polynomials Ψ_i.

$$P+1 = 1+\sum_{s=1}^{d}\frac{1}{s!}\prod_{r=0}^{s-1}(n+r) = 1+\sum_{s=1}^{d}\frac{(n-1+s)!}{(n-1)!s!} = \frac{(n+d)!}{n!d!} \tag{3.11}$$

Equation 3.11 shows clearly that as n or d increases, the number of the expansion terms increases rapidly, which is called the **Curse of Dimensionality** [29]. This will impose some practical limitations on the efficiency of numerical solution by the generalized PCE.

3.2.4 Statistical Moments based on the Generalized PCE

Given the truncated expansion as shown in Equation 3.10, for UQ purposes we would like to know the mean and variance of a certain random variable X. The mean of X is:

$$\begin{aligned}\bar{X} = \mathbb{E}[X(\boldsymbol{\xi})] &= \int\sum_{i=0}^{P}x_i\Psi_i(\boldsymbol{\xi})p(\boldsymbol{\xi})d\boldsymbol{\xi} = \sum_{i=0}^{P}x_i\int\Psi_i(\boldsymbol{\xi})p(\boldsymbol{\xi})d\boldsymbol{\xi}\\ &= \sum_{i=0}^{P}x_i\int\Psi_i(\boldsymbol{\xi})\Psi_0(\boldsymbol{\xi})p(\boldsymbol{\xi})d\boldsymbol{\xi} = \sum_{i=0}^{P}x_i\langle\Psi_0,\Psi_i\rangle = x_0\end{aligned} \tag{3.12}$$

The variance of the solution is obtained by:

$$\begin{aligned}\mathrm{Var}(X) = \mathbb{E}\left\langle X^2(\boldsymbol{\xi}) - x_0^2 \right\rangle &= \mathbb{E}\left\langle \sum_{i=0}^{P}\sum_{k=0}^{P} x_i\Psi_i(\boldsymbol{\xi})x_k\Psi_k(\boldsymbol{\xi}) \right\rangle - x_0^2 \\ &= \sum_{i=0}^{P}\sum_{k=0}^{P} x_i x_k \left\langle \Psi_i, \Psi_k \right\rangle \delta_{ik} - x_0^2 = \sum_{k=1}^{P} x_k^2 \left\langle \Psi_k^2 \right\rangle \end{aligned} \tag{3.13}$$

Therefore, the mean value is the expansion coefficient with index zero. The variance only depends on the rest expansion coefficients since $\left\langle \Psi_k^2 \right\rangle$ is known analytically once the orthogonal polynomials are chosen.

For generalized PCE as shown in Equation 3.10, the orthogonal polynomials $\Psi_i(\boldsymbol{\xi})$ are chosen by the user according to the type of random distribution of $X(\boldsymbol{\xi})$. The correspondence of different random distribution and different family of generalized orthogonal polynomials can be found in Table 3.1. The expansion coefficients are the unknowns. We need methods to solve for these unknown modes to get a closed solution of PCE. Available methods are intrusive Galerkin Projection, NISP and regression-based methods. In the following we will only briefly introduce the intrusive Galerkin projection method.

3.2.5 Intrusive Galerkin Projection

The Galerkin projection method is based on the orthogonality of the polynomials:

$$\left\langle \Psi_i(\boldsymbol{\xi}), \Psi_j(\boldsymbol{\xi}) \right\rangle = \int \Psi_i(\boldsymbol{\xi})\Psi_j(\boldsymbol{\xi}) dp(\boldsymbol{\xi}) = \int \Psi_i(\boldsymbol{\xi})\Psi_j(\boldsymbol{\xi}) w(\boldsymbol{\xi}) d\xi = \delta_{ij} \left\langle \Psi_i^2 \right\rangle$$

Galerkin projection consists of pre-multiplying Equation 3.10 by $\Psi_k(\boldsymbol{\xi})$ and taking the expectation of both sides:

$$\begin{aligned}\mathbb{E}\left[X\Psi_k(\boldsymbol{\xi})\right] = \left\langle X, \Psi_k(\boldsymbol{\xi}) \right\rangle = \left\langle \sum_{i=0}^{P} x_i\Psi_i(\boldsymbol{\xi}), \Psi_k(\boldsymbol{\xi}) \right\rangle &= \mathbb{E}\left[\sum_{i=0}^{P} x_i \Psi_i \Psi_k\right] \\ = \sum_{i=0}^{P} x_i \left\langle \Psi_i, \Psi_k \right\rangle = \sum_{i=0}^{P} x_i \left\langle \Psi_i^2 \right\rangle \delta_{ik} &= x_k \left\langle \Psi_k^2 \right\rangle \end{aligned} \tag{3.14}$$

$$x_k = \frac{\mathbb{E}\left[X\Psi_k(\boldsymbol{\xi})\right]}{\left\langle \Psi_k^2 \right\rangle} = \frac{\left\langle X, \Psi_k \right\rangle}{\left\langle \Psi_k^2 \right\rangle}, \qquad k = 0, 1, 2, \ldots \tag{3.15}$$

Next, consider a stochastic Ordinary Differential Equation (ODE) or Partial Differential Equation (PDE):

$$\mathcal{L}\left(\mathbf{x}, t, u; \omega\right) = f\left(\mathbf{x}, t; \omega\right) \tag{3.16}$$

where $u = u(\mathbf{x},t;\omega)$ is the solution and $f(\mathbf{x},t;\omega)$ is the source term. Operator $\mathcal{L}$ in the differential equation generally involves differentiations in space/time and can be nonlinear. Appropriate initial and boundary conditions are assumed. The existence of randomness ω is due to the introduction of uncertainty into the system via BCs, ICs, material properties or parameters in the differential equation.

The solution u, which is treated as a random process, can be expanded using the generalized PCE as:

$$u(\mathbf{x},t;\omega) = \sum_{i=0}^{P} u_i(\mathbf{x},t)\,\Phi_i(\boldsymbol{\xi}(\omega)) \tag{3.17}$$

The source term is also random and can be expanded in a similar way:

$$f(\mathbf{x},t;\omega) = \sum_{i=0}^{P} f_i(\mathbf{x},t)\,\Phi_i(\boldsymbol{\xi}(\omega)) \tag{3.18}$$

Next, we substitute the expansion of $u(\mathbf{x},t;\omega)$ and $f(\mathbf{x},t;\omega)$ into the original differential equation:

$$\mathcal{L}\left(\mathbf{x},t,\sum_{i=0}^{P} u_i\Phi_i\right) = \sum_{i=0}^{P} f_i\Phi_i \tag{3.19}$$

Then, a Galerkin projection of the above equation is done onto each polynomial basis Φ_i in order to ensure that the approximation error is orthogonal to the functional space spanned by the finite-dimensional basis Φ_k. For $k = 0,1,2,\dots,P$, we have:

$$\left\langle \mathcal{L}\left(\mathbf{x},t,\sum_{i=0}^{P} u_i\Phi_i\right),\Phi_k\right\rangle = \left\langle \sum_{i=0}^{P} f_i\Phi_i,\Phi_k\right\rangle = \sum_{i=0}^{P} f_i\left\langle\Phi_i^2\right\rangle\delta_{ik} = f_k\left\langle\Phi_k^2\right\rangle \tag{3.20}$$

Using the orthogonality of the polynomial basis, we obtain a set of $(1+P)$ coupled equations for each random mode $u_i(\mathbf{x},t)$ where $i = \{0,1,2,\dots,P\}$. It should be noted that by utilizing the generalized PCE, the randomness is effectively transferred into the basis polynomials. Thus, the governing equations for the expansion coefficients $\{u_i\}_{i=0}^{P}$ resulted from the above equation are deterministic. Discretization in space $\mathbf{x}$ and time t can be carried out by any conventional deterministic techniques.

3.3 Stochastic Collocation

SC and PCE both belong to a large class of UQ methods called stochastic spectral techniques. PCE estimates coefficients for known orthogonal polynomial basis functions, whereas SC forms interpolation functions for known coefficients [31] (which are function evaluations).

There are several advantages of using SC over PCE to solve PDEs/ODEs: (1) SC results in series of uncoupled deterministic sub-problems for which legacy code can be used essentially unmodified; (2) unlike MC sampling, SC can take advantage of the smooth dependence of the solution on the random parameters to yield spectral convergence; (3) nonlinear problems pose no additional difficulty (unlike intrusive Galerkin projection used to solve for PCE).

The basic idea of the SC technique is to approximate the multi-dimensional stochastic space of the problem $f(x)$ with interpolation functions [31] [160] at a set of collocation points $\{x_i\}_{i=1}^{d}$. Deterministic solutions of the problem at each point x_i are used to construct an interpolant of $f(x)$ by using linear combinations of $f(x_i)$. There are two choices for such multi-dimensional interpolation; full tensor product of one-dimensional interpolation rules or sparse grid interpolation rules based on the Smolyak algorithm [31] [127]. Sparse grid can be used for both numerical integration and interpolation, but in the following we will use interpolation for illustration.

Suppose we want to approximate smooth functions $f : [-1,1]^d \to \mathbb{R}$, using a finite number of function values. For a one-dimensional problem we use:

$$\mathcal{U}^i(f) = \sum_{j=1}^{m_i} f(x_j^i) \cdot \alpha_j^i \tag{3.21}$$

where $i \in \mathbb{N}$ denotes that the rule is in the i^{th} dimension. m_i is the number of nodes. $\{\alpha_j^i\}_{j=1}^{m_i} \in C([-1,1])$ are the Lagrange polynomials for interpolation and $\{x_j^i\}_{j=1}^{m_i} \in [-1,1]$ are nodes in the i^{th} dimension. Finally, $\{f(x_j^i)\}_{j=1}^{m_i}$ are evaluation of function at these interpolation nodes.

We assume that a sequence of formulas is given for $\{i = 1,2,...,d\}$. For the multivariate case $d > 1$ we first define tensor product formulas:

$$\left(\mathcal{U}^{i_1} \otimes \cdots \otimes \mathcal{U}^{i_d}\right)(f) = \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} f\left(x_{j_1}^{i_1}, \cdots, x_{j_d}^{i_d}\right) \cdot \left(\alpha_{j_1}^{i_1} \otimes \cdots \otimes \alpha_{j_d}^{i_d}\right) \tag{3.22}$$

which serves as building blocks for the Smolyak algorithm [127]. The above product formula needs $\left(\prod_{j=1}^{d} m_{i_j}\right)$ function evaluations. If we have the same number of nodes m_i in each dimension then the total number of nodes will be $(m_i)^d$. The number of nodes grows exponentially as the number of dimensions d increases and quickly exceeds the available computational power. Therefore, like generalized PCE, SC also suffers from the "curse of dimensionality".

The **Smolyak algorithm** $\mathcal{A}(q,d)$, where q is called the level of the sparse grid that is independent of the dimension d, is a linear combination of product formulas in one dimension with the following key properties. Only products with a relatively small number of nodes are used and the linear combination is chosen in such a way that the interpolation property for $d = 1$ is preserved for $d > 1$ [8] [46].

For $i \in \mathbb{N}$, define the difference operator:

$$\mathcal{U}^0(f) = 1 \tag{3.23}$$

$$\Delta^i(f) = \mathcal{U}^i(f) - \mathcal{U}^{i-1}(f) \tag{3.24}$$

Moreover, define $|\mathbf{i}| = i_1 + i_2 + \dots i_d$ for $i \in \mathbb{N}^d$. Then, the Smolyak algorithm $\mathcal{A}(q,d)$ is given by:

$$\mathcal{A}(q,d)(f) = \sum_{|\mathbf{i}| \leq q+d} \left(\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d}\right)(f) \tag{3.25}$$

Another form of the Smolyak algorithm is:

$$\mathcal{A}(q,d)(f) = \sum_{q+1 \leq |\mathbf{i}| \leq q+d} (-1)^{q+d-|\mathbf{i}|} \binom{d-1}{q+d-|\mathbf{i}|} \cdot \left(\mathcal{U}^{i_1} \otimes \cdots \otimes \mathcal{U}^{i_d}\right)(f) \tag{3.26}$$

To compute $\mathcal{A}(q,d)(f)$, one only needs to know function values at the "sparse grid" defined as:

$$\mathcal{H}(q,d) = \bigcup_{q+1 \leq |\mathbf{i}| \leq q+d} \left(\mathcal{X}^{i_1} \otimes \cdots \otimes \mathcal{X}^{i_d}\right) \tag{3.27}$$

where $\mathcal{X}^{i_k} = \left\{x_{j_1}^{i_k}, x_{j_2}^{i_k}, \dots, x_{j_d}^{i_k}\right\} \subset [-1,1]$ denotes the set of interpolation nodes used by $\mathcal{U}^{i_k}$.

To build the sparse grid, one should start from one-dimensional interpolation/integration rules (for example, Gauss rules and Clenshaw-Curtis rules). Figure 3.2 shows the comparison of full tensor grids and sparse grids using Clenshaw-Curtis rule for isotropic grids of level 4 and 5. Full tensor grids include 289 and 1089 nodes for levels 4 and 5 respectively, while sparse grids only need 65 and 145 nodes respectively. The sparse grid is more advantageous with increasing number of levels, as the reduction of nodes becomes significant.

The grid of nodes is considered isotropic when an equal number of nodes is assigned for each dimension. Although the number of nodes increase at the rate m^d for isotropic full tensor grids, the rate of increase is $m^{\log(d)}$ for isotropic sparse grids [31] [160]. When the dimension is very high ($d \geq 20$), even a sparse grid requires a large number of nodes. Dimension-adaptive sparse grids have been used based on the fact that not all input parameters are equally important to the output [47]. For dimensions where the output is very smooth, less nodes can be used to alleviate the computational burden. See [152] [156]for a few examples of SGSC and their applications.

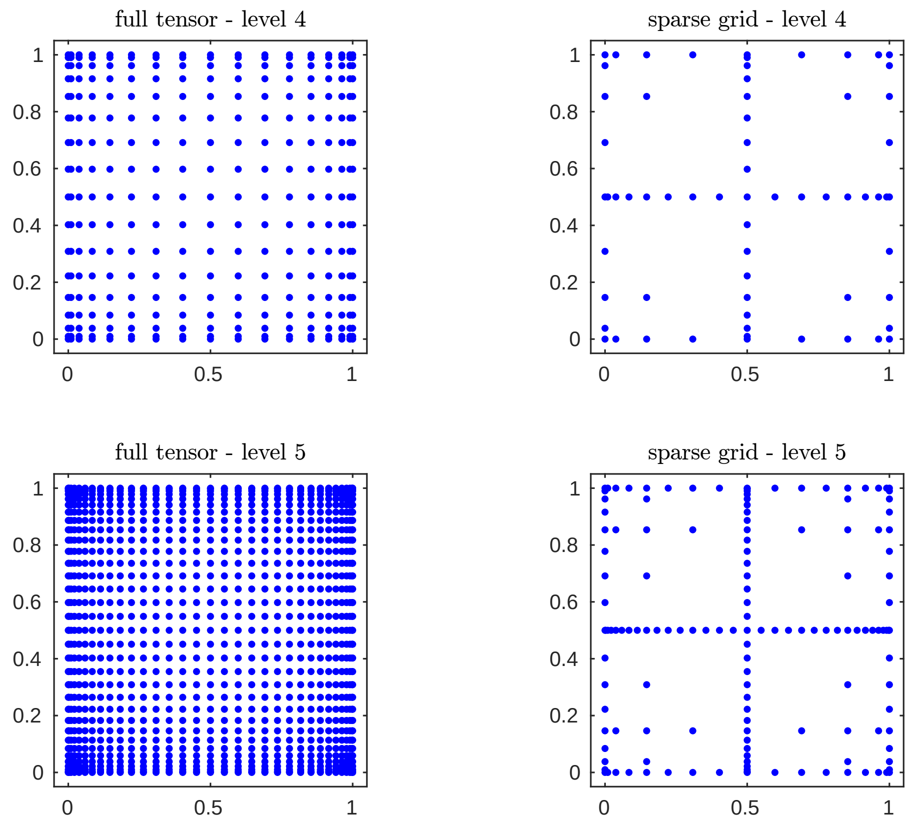


Fig. 3.2 Comparison of full tensor grid and classical isotropic sparse grid based on Clenshaw-Curtis rule.

3.4 Summary and Comments

Both intrusive and non-intrusive approaches are more efficient than brute-force MC sampling up to very high random dimensionality. The main disadvantage of Galerkin-PCE method is that it requires a large new coding effort. It results in a new coupled system of equations that is P times larger than the original system of equations, P increase quickly with the dimension of random input parameters and the order of PCE. Galerkin-PCE approach can also be challenging when the governing equations take complicated forms. In this case, the derivation of explicit equations for the generalized PCE coefficients can be very difficult, if not impossible.

On the other hand, NISP techniques as non-intrusive methods rely only on the definition of the sampling points in the random space. Consequently, the extension of NISP to large scale problems is straightforward. As long as the quadrature points are chosen, an existing simulation code can be used as "black box" for the calculation of their values. SC methods generally result in a larger number of equations than a typical Galerkin-PCE method. However, these equations are completely decoupled, which makes them easier to solve and require only multiple runs of a deterministic solver. Therefore, for problems with complicated governing equations, SC methods should perform better. A SC algorithm is similar to MC sampling approach in that only repetitive realization of a deterministic solver are required by choose a set of nodes. However, based on the theory of multivariate polynomial interpolations, it also possesses the nature of high accuracy and fast convergence of Galerkin-PCE.

Once the PCE coefficients are solved, the model output is approximated as a polynomial function of random input parameters, which can be used to describe stochastic nature of the output in terms of its mean, variance, covariance, distribution, etc. The PCE essentially constitutes a meta-model of the original problem and can be used as surrogates during the inverse UQ process. There are four steps in Bayesian inference of inverse UQ using metamodel constructed by PCE:

1. Construct generalized PCE for each random input parameter, according to prior uncertainty distributions. Solve for the PCE coefficients;
2. Substitute these expansions (with known PCE coefficients) into the governing equations of the full model and use Galerkin projection to obtain a new coupled system of equations for the generalized PCE coefficients. The solution for the new system of equations will be the PCE coefficients for QoIs.;
3. Solve this system of equations to get the PCE modes;
4. Formulate the posterior with available experimental data and PCE metamodel;
5. Explore this posterior with MCMC sampling strategy.

Chapter 4

GAUSSIAN PROCESS MODELING

Gaussian Process (GP) modeling, also known as **Kriging**, or spatial correlation modeling [74], was originally developed by geologists in the 1950s to predict distribution of minerals over an area of interest given a set of sampled sites [24] [25] [90] [130]. It was made popular in the context of modeling and optimization by Sacks et al. [26] [119] and Jones et al. [69] respectively. GP has been widely used to construct metamodels for deterministic computer models in many areas [74] [86]. A GP model is a generalized linear regression model that accounts for the correlation in the residuals between the regression model and the observations. Here the "observations" are realizations of the computer code at selected locations of the inputs, rather than experimental data. The only assumption for GP modeling is that the model QoI is continuous and smooth over the input domain [38], which is true for most computer models. We thus believe that if two points are close to each other in the input domain, the residuals in the regression model should be close [69]. It follows that we do not treat the residuals as independent, assume that the correlation between the residuals are related to the distance between the corresponding input points.

In this section, we will present a self-contained introduction of GP, including the general theory, correlation kernels, prediction and Mean Square Error (MSE) formula, design of computer experiments, parameter estimation, accuracy assessment and implementations issues. However, we understand that the richness of the GP modeling filed is just hinted in this section, given the fact that several books have been devoted on this topic [24] [25] [90] [115] [121] [130]. But we hope that the process to use GP metamodeling technique can be made apparent to the nuclear engineering community.

4.1 GP General Theory

4.1.1 Mathematical Formulation

Consider a mathematical model of the general form $y = y^{\mathrm{M}}(\mathbf{x})$ (not to be confused with the definition of computer model in Chapter 2, here $\mathbf{x}$ include both design and calibration parameters). Without loss of generality, assume that y is a scalar and $\mathbf{x} = [x_1, x_2, \ldots, x_d]$ representing d input parameters. Assume that the computer model output is known at m design sites (also called training sites):

$$\mathbf{X} = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}\right]^{\top} \tag{4.1}$$

where $\mathbf{X}$ is a $m \times d$ design matrix. The corresponding output values are:

$$\mathbf{y} = [y_1, y_2, \ldots, y_m]^{\top} = \left[y^{\mathrm{M}}(\mathbf{x}^{(1)}), y^{\mathrm{M}}(\mathbf{x}^{(2)}), \ldots, y^{\mathrm{M}}(\mathbf{x}^{(m)})\right]^{\top} \tag{4.2}$$

The mathematical form of a GP model has two parts:

$$y(\mathbf{x}) = \sum_{j=1}^{n} \beta_j f_j(\mathbf{x}) + z(\mathbf{x}) = \mathbf{f}^{\top}(\mathbf{x})\boldsymbol{\beta} + z(\mathbf{x}) \tag{4.3}$$

Here $y(\mathbf{x})$ is the output prediction at a general location denoted by $\mathbf{x}$. The first part is a linear regression of the data with n regressors modeling the drift of the mean, also called the "**trend**". The set of **basis functions** $\mathbf{f} = [f_1, f_2, \ldots, f_n]^{\top}$ is known and chosen by the user, while the vector $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_n]^{\top}$ contains the unknown **regression coefficients**. The second part $z(\mathbf{x})$ is a **stationary Gaussian random process** with zero mean and covariance:

$$\mathrm{Cov}\left[z\left(\mathbf{x}^{(i)}\right), z\left(\mathbf{x}^{(j)}\right)\right] = \sigma^2 \mathcal{R}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) \tag{4.4}$$

where σ^2 is a scalar parameter called **process variance** of GP and $\mathcal{R}(\cdot,\cdot)$ is called the **correlation function** or **correlation kernel**. The inclusion of the second part $z(\mathbf{x})$ generalizes the linear regression model to a "*stochastic process model*".

4.1.2 Correlation Kernels

An arbitrary function for $\mathcal{R}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)$ is generally not valid unless the following conditions are satisfied:

- The correlation function should be symmetric, i.e., for two arbitrary points $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ in the input domain:

$$\mathcal{R}\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \mathcal{R}\left(\mathbf{x}^{(j)},\mathbf{x}^{(i)}\right)$$

- The resulting correlation matrix should be a positive semi-definite matrix.

The correlation kernel is often chosen to be a function of the distance:

$$\mathcal{R}\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \mathcal{R}\left[d\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right)\right] \tag{4.5}$$

Instead of using the Euclidean distance that weights all the variables equally, we use a specially weighted distance formula defined below:

$$d\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right) = \sum_{k=1}^{d} \frac{|x_k^{(i)} - x_k^{(j)}|^{p_k}}{\theta_k} \tag{4.6}$$

For example, if we choose the power-exponential function for the correlation kernel $\mathcal{R}(\cdot,\cdot)$, the covariance becomes:

$$\mathrm{Cov}\left[z(\mathbf{x}^{(i)}), z(\mathbf{x}^{(j)})\right] = \sigma^2 \cdot \exp\left[-\sum_{k=1}^{d} \frac{|x_k^{(i)} - x_k^{(j)}|^{p_k}}{\theta_k}\right] \tag{4.7}$$

The unknown parameters $\mathbf{p} = [p_1, p_2, \ldots, p_d]$ are called **roughness parameters**, which control the smoothness of the correlation function. It is usually required that $p_k \in [1,2]$ [69]. Note that some others adopted a wider range $p_k \in [0,2]$ [37] [104] [118]. It can be seen from Figure 4.1 (left) that when the distance h between two points approaches 0, the correlation gets closer to 1. And the correlation decreases when the distance gets larger. The value $p = 2$ corresponds to smooth functions with a continuous gradient at $h = 0$ and p values near 1 or less correspond to less smoothness.

The unknown parameters[1] $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_d]$ ($\theta_k \geq 0$) are called **characteristic length-scales** [115], which are essentially width parameters that control how far a sample point's influence extends [38]. As shown in Figure 4.1 (right), given $p = 2$ and at the same distance, larger θ_k values result in higher correlation, indicating that output values are close. Smaller θ_k values lead to lower correlation, meaning that there will be large difference between outputs even for nearby points.

[1]Note that we use symbol $\boldsymbol{\theta}$ here to be consistent with the previous literature. $\boldsymbol{\theta}$ represents the characteristic length-scales only in this chapter. In other chapters, $\boldsymbol{\theta}$ still stands for the calibration parameters while the characteristic length-scales will be represented by $\boldsymbol{\theta}$ with certain superscripts.

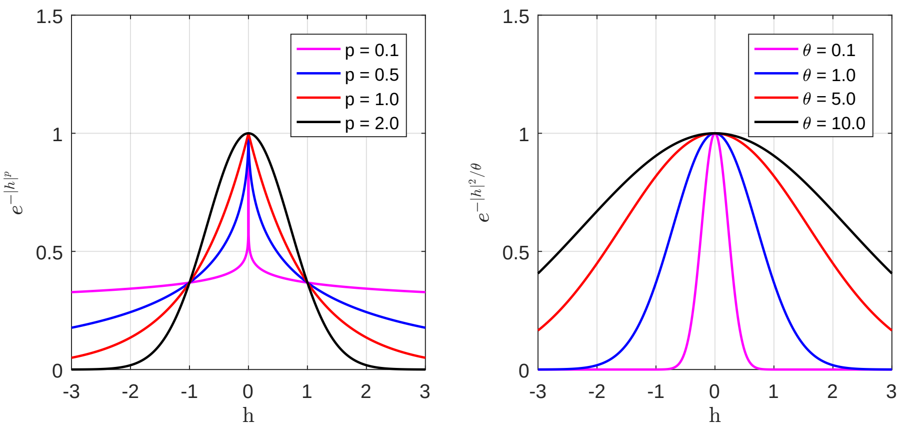


Fig. 4.1 Demonstration of the correlation with different values of roughness parameter p (left) and characteristic length scale θ (right). The x-axis h means the distance between two points in one-dimension. The left subfigure fixes $\theta = 1$ while the right subfigure fixes $p = 2$.

Table 4.1 Common spatial correlation kernels (the distance h and correlation kernel formulas are defined for each dimension for notational clarity).

Name	Expression $\left(h = \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|\right)$
Linear	$\mathcal{R}(h) = \max\left(0, 1 - \frac{\|h\|}{\theta}\right)$
Exponential	$\mathcal{R}(h) = \exp\left(-\frac{\|h\|}{\theta}\right)$
Power-exponential	$\mathcal{R}(h) = \exp\left(-\left(\frac{\|h\|}{\theta}\right)^p\right)$
Gaussian	$\mathcal{R}(h) = \exp\left(-\frac{\|h\|^2}{2\theta^2}\right)$
Matérn $\nu = 3/2$	$\mathcal{R}(h) = \left(1 + \frac{\sqrt{3}\|h\|}{\theta}\right)\exp\left(-\frac{\sqrt{3}\|h\|}{\theta}\right)$
Matérn $\nu = 5/2$	$\mathcal{R}(h) = \left(1 + \frac{\sqrt{5}\|h\|}{\theta} + \frac{5h^2}{3\theta^2}\right)\exp\left(-\frac{\sqrt{5}\|h\|}{\theta}\right)$

From the above discussion, apparently the choice of the correlation kernel is crucial to the performance of GP metamodel. It is required that the kernel is positive definite. A commonly accepted approach is to select the correlation kernel beforehand from kernels that are known to be positive definite. Table 4.1 shows some commonly used correlation kernels. Note that the parameter θ can be different for each of the d dimensions. Among these kernels, Matérn family kernels have been suggested by statisticians, while engineers often favour the Gaussian (or square-exponential) kernel [86]. The latter results in a smooth and infinitely differentiable function which is desirable for many engineering applications. For some other detailed discussion of the correlation kernels, see [38] [77] [115].

4.1.3 Prediction and Mean Square Error Formulas

Given the design matrix $\mathbf{X}$ and corresponding output values $\mathbf{y}$, to predict the output value at a new input location $\mathbf{x}^*$ using the GP model, we first define the following:

1. The set of regression functions $\mathbf{f} = [f_1, f_2, \dots, f_n]^\top$ evaluated at the untried point $\mathbf{x}^*$:

$$\mathbf{f}(\mathbf{x}^*) = [f_1(\mathbf{x}^*), f_2(\mathbf{x}^*), \dots, f_n(\mathbf{x}^*)]^\top \tag{4.8}$$

2. The set of regression functions evaluated at the m design sites, which is also called the *information matrix*:

$$\mathbf{F} = \left[\mathbf{f}\left(\mathbf{x}^{(1)}\right), \mathbf{f}\left(\mathbf{x}^{(2)}\right), \dots, \mathbf{f}\left(\mathbf{x}^{(m)}\right)\right]^\top = \begin{bmatrix} f_1(\mathbf{x}^{(1)}) & f_2(\mathbf{x}^{(1)}) & \cdots & f_n(\mathbf{x}^{(1)}) \\ f_1(\mathbf{x}^{(2)}) & f_2(\mathbf{x}^{(2)}) & \cdots & f_n(\mathbf{x}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(\mathbf{x}^{(m)}) & f_2(\mathbf{x}^{(m)}) & \cdots & f_n(\mathbf{x}^{(m)}) \end{bmatrix} \tag{4.9}$$

3. The correlation of $\mathbf{x}^*$ with the m design points:

$$\mathbf{r}(\mathbf{x}^*) = \left[\mathcal{R}\left(\mathbf{x}^*, \mathbf{x}^{(1)}\right), \mathcal{R}\left(\mathbf{x}^*, \mathbf{x}^{(2)}\right), \dots, \mathcal{R}\left(\mathbf{x}^*, \mathbf{x}^{(m)}\right)\right]^\top \tag{4.10}$$

4. The correlation between the m design points:

$$\mathbf{R} = \begin{bmatrix} \mathcal{R}\left(\mathbf{x}^{(1)}, \mathbf{x}^{(1)}\right) & \mathcal{R}\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right) & \cdots & \mathcal{R}\left(\mathbf{x}^{(1)}, \mathbf{x}^{(m)}\right) \\ \mathcal{R}\left(\mathbf{x}^{(2)}, \mathbf{x}^{(1)}\right) & \mathcal{R}\left(\mathbf{x}^{(2)}, \mathbf{x}^{(2)}\right) & \cdots & \mathcal{R}\left(\mathbf{x}^{(2)}, \mathbf{x}^{(m)}\right) \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{R}\left(\mathbf{x}^{(m)}, \mathbf{x}^{(1)}\right) & \mathcal{R}\left(\mathbf{x}^{(m)}, \mathbf{x}^{(2)}\right) & \cdots & \mathcal{R}\left(\mathbf{x}^{(m)}, \mathbf{x}^{(m)}\right) \end{bmatrix} \tag{4.11}$$

The dimensions and descriptions of all the symbols are summarized in Table 4.2. The prediction, or mean value of the output at the untried input $\mathbf{x}^*$ is:

$$\hat{y}(\mathbf{x}^*) = \mu_{\hat{y}}(\mathbf{x}^*) = \mathbf{f}^\top(\mathbf{x}^*)\widehat{\boldsymbol{\beta}} + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{F}\widehat{\boldsymbol{\beta}}\right) \tag{4.12}$$

where $\widehat{\boldsymbol{\beta}}$ is the least squares estimate of the regression coefficients $\boldsymbol{\beta}$, also called the **Best Linear Unbiased Estimator (BLUE)**:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{y} \tag{4.13}$$

Table 4.2 Symbols used in GP metamodel formulation

Symbol	Dimension	Description
d	1×1	number of input factors
m	1×1	number of design points
n	1×1	number of basis functions
$\boldsymbol{\beta}$	$n\times 1$	vector of regression coefficients
$\widehat{\boldsymbol{\beta}}$	$n\times 1$	estimator of $\boldsymbol{\beta}$
$\mathbf{f}$	$n\times 1$	vector of regression functions $\mathbf{f}=[f_1,f_2,\dots,f_n]^\top$
$\mathbf{x}^{(i)}$	$d\times 1$	vector of i^{th} design point, $\mathbf{x}^{(i)}=\left(x_1^{(i)},x_2^{(i)},\dots,x_d^{(i)}\right)^\intercal$
y_i	1×1	output value of i^{th} design point, $y_i=y^{\text{M}}(\mathbf{x}^{(i)})$
$\mathbf{X}$	$m\times d$	collection of m design points, $\mathbf{X}=\left[\mathbf{x}^{(1)},\mathbf{x}^{(2)},\dots,\mathbf{x}^{(m)}\right]^\top$
$\mathbf{y}$	$m\times 1$	vector of output values at $\mathbf{X}$, $\mathbf{y}=(y_1,y_2,\dots,y_m)^\top$
$\mathbf{x}^*$	$d\times 1$	untried input point, to be predicted
$\mathbf{f}(\mathbf{x}^*)$	$n\times 1$	$\mathbf{f}$ evaluated at untried location $\mathbf{x}^*$
$\mathbf{F}$	$m\times n$	$\mathbf{f}$ evaluated at design sites $\mathbf{X}$
$\mathbf{r}(\mathbf{x}^*)$	$m\times 1$	correlation between $\mathbf{x}^*$ and design sites $\mathbf{X}$
$\mathbf{R}$	$m\times m$	correlation between design sites $\mathbf{X}$
$\hat{y}(\mathbf{x}^*)$ or $\mu_{\hat{y}}(\mathbf{x}^*)$	1×1	predicted output at $\mathbf{x}^*$
$\text{MSE}\,[\hat{y}(\mathbf{x}^*)]$ or $\sigma^2_{\hat{y}}(\mathbf{x}^*)$	1×1	variance of the prediction at $\mathbf{x}^*$

The predictor $\hat{y}(\mathbf{x}^*)$ in Equation 4.12 can be proven to be the **Best Linear Unbiased Predictor (BLUP)** [119] [121]. The MSE or variance of the predictor $\hat{y}(\mathbf{x}^*)$ is:

$$\text{MSE}\,[\hat{y}(\mathbf{x}^*)]=\sigma^2_{\hat{y}}(\mathbf{x}^*)=\sigma^2\left[1-\begin{bmatrix}\mathbf{f}^\top(\mathbf{x}^*) & \mathbf{r}^\top(\mathbf{x}^*)\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{F}^\top\\ \mathbf{F} & \mathbf{R}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{f}(\mathbf{x}^*)\\ \mathbf{r}(\mathbf{x}^*)\end{bmatrix}\right] \tag{4.14}$$

Equation 4.14 can be expanded in another popular form for the MSE:

$$\begin{aligned}\text{MSE}\,[\hat{y}(\mathbf{x}^*)]=\sigma^2_{\hat{y}}(\mathbf{x}^*)=\sigma^2\Big[1-\mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)+&\\ \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)-\mathbf{f}(\mathbf{x}^*)\right)^\top\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)-\mathbf{f}(\mathbf{x}^*)\right)\Big]&\end{aligned} \tag{4.15}$$

Detailed derivation of the predictor and MSE can be found in Appendix F. Because GP assumes that the output at untried input $\mathbf{x}^*$ follows a Gaussian distribution, in the following we will use the mean value $\mu_{\hat{y}}(\mathbf{x}^*)$ and variance $\sigma^2_{\hat{y}}(\mathbf{x}^*)$ to represent the predictor and MSE.

4.1.4 Different Types of GP

In the literature, different naming of Kriging are defined depending on the trend functions used:

1. Equation 4.12 - 4.15 formulate the most general and flexible case of Kriging model, named **Universal Kriging (UK)**. UK uses a linear combination of prescribed functions, e.g.polynomials, as the trend.

2. **Ordinary Kriging (OK)** is a special case of UK when the basis functions reduce to [1], while the trend has a constant yet unknown value β_0:

$$\mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta} = \beta_0 \tag{4.16}$$

$$\widehat{\beta_0} = \frac{\mathbf{1}_m^\top \mathbf{R}^{-1}\mathbf{y}}{\mathbf{1}_m^\top \mathbf{R}^{-1}\mathbf{1}_m} \tag{4.17}$$

$$\mu_{\hat{y},\mathrm{OK}}(\mathbf{x}^*) = \widehat{\beta_0} + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{1}_m\widehat{\beta_0}\right) \tag{4.18}$$

$$\sigma^2_{\hat{y},\mathrm{OK}}(\mathbf{x}^*) = \sigma^2\left[1 - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) + \frac{\left[1 - \mathbf{1}_m^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)\right]^2}{\mathbf{1}_m^\top\mathbf{R}^{-1}\mathbf{1}_m}\right] \tag{4.19}$$

   where $\mathbf{1}_m$ is a length-m column vector of ones.

3. **Simple Kriging (SK)** is the case when the trend is a known constant value μ:

$$\mu_{\hat{y},\mathrm{SK}}(\mathbf{x}^*) = \mu + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}_m\mu) \tag{4.20}$$

$$\sigma^2_{\hat{y},\mathrm{SK}}(\mathbf{x}^*) = \sigma^2\left[1 - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)\right] \tag{4.21}$$

It is obvious that OK and SK are special cases of UK. The most commonly used basis functions are constant, linear and quadratic.

$$\mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta} = \beta_0$$

$$\mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta} = \beta_0 + \sum_{j=1}^{n}\beta_j x_j$$

$$\mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta} = \beta_0 + \sum_{j=1}^{n}\beta_j x_j + \sum_{j_1=1}^{n}\sum_{j_2=1}^{n}\beta_{j_1 j_2} x_{j_1} x_{j_2}$$

In theory one can choose polynomials of arbitrary order for the trend. However, it turns out unnecessary for most engineering problems. Because GP is an interpolator (to be shown later), trend functions have little effect within the design domain. They only have major influence beyond the design sites when GP is used as extrapolator. Furthermore, we do not have a prior knowledge of trend of the computer model. Arbitrarily specifying the trend function may introduce inaccuracies. In fact, OK is the most widely used version of GP.

4.1.5 Interpolation Property

One important property of GP emulator is that the mean prediction is a linear combination of basis functions shown in Equation 4.12, which interpolates the design sites. Furthermore, The predicted variance increases as new point get further away from design sites.

To prove that the GP metamodel interpolates the design sites, we need to show that $\mu_{\hat{y}}(\mathbf{x}^{(i)}) = y_i$ and $\sigma^2_{\hat{y}}(\mathbf{x}^*) = 0$. Let $\mathbf{x}^* = \mathbf{x}^{(i)}$, which is one of the design sites. $\mathbf{f}^\top(\mathbf{x}^*)$ becomes the i^{th} row of the information matrix $\mathbf{F}$:

$$\mathbf{f}^\top(\mathbf{x}^*) = \mathbf{f}^\top(\mathbf{x}^{(i)}) = \left[f_1(\mathbf{x}^{(i)}), f_2(\mathbf{x}^{(i)}), \ldots, f_n(\mathbf{x}^{(i)})\right] = \mathbf{F}_{i\cdot} \tag{4.22}$$

$\mathbf{r}^\top(\mathbf{x}^*)$ becomes the i^{th} row of the correlation matrix $\mathbf{R}$:

$$\mathbf{r}^\top(\mathbf{x}^*) = \mathbf{r}^\top(\mathbf{x}^{(i)}) = \left[\mathcal{R}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(1)}\right), \mathcal{R}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(2)}\right), \ldots, \mathcal{R}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(m)}\right)\right] = \mathbf{R}_{i\cdot} \tag{4.23}$$

Given $\mathbf{e}_i$ as the unit column vector in the i^{th} dimension, we have:

$$\mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1} = \mathbf{R}_{i\cdot}\mathbf{R}^{-1} = \mathbf{e}_i^\top \tag{4.24}$$

The predictor in Equation 4.12 becomes:

$$\begin{aligned} \mu_{\hat{y}}(\mathbf{x}^*) &= \mathbf{f}^\top(\mathbf{x}^*)\widehat{\boldsymbol{\beta}} + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{F}\widehat{\boldsymbol{\beta}}\right) \\ &= \mathbf{F}_{i\cdot}\widehat{\boldsymbol{\beta}} + \mathbf{e}_i^\top\left(\mathbf{y} - \mathbf{F}\widehat{\boldsymbol{\beta}}\right) = \mathbf{F}_{i\cdot}\widehat{\boldsymbol{\beta}} + y_i - \mathbf{F}_{i\cdot}\widehat{\boldsymbol{\beta}} \\ &= y_i \end{aligned} \tag{4.25}$$

Also:

$$\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*) = \mathbf{F}^\top\mathbf{R}^{-1}\mathbf{R}_{i\cdot}^\top - \mathbf{F}_{i\cdot}^\top = \mathbf{F}^\top\mathbf{e}_i - \mathbf{F}_{i\cdot}^\top = \mathbf{0} \tag{4.26}$$

The MSE in Equation 4.15 becomes:

$$\begin{aligned} \sigma^2_{\hat{y}}(\mathbf{x}^*) &= \sigma^2_{\hat{y}}(\mathbf{x}^{(i)}) \\ &= \sigma^2\left[1 - \mathbf{e}_i^\top\mathbf{R}_{i\cdot}^\top + \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{R}_{i\cdot}^\top - \mathbf{F}_{i\cdot}^\top\right)^\top\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{R}_{i\cdot}^\top - \mathbf{F}_{i\cdot}^\top\right)\right] \\ &= \sigma^2\left[1 - 1 + \mathbf{0}^\top\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{0}\right] = 0 \end{aligned} \tag{4.27}$$

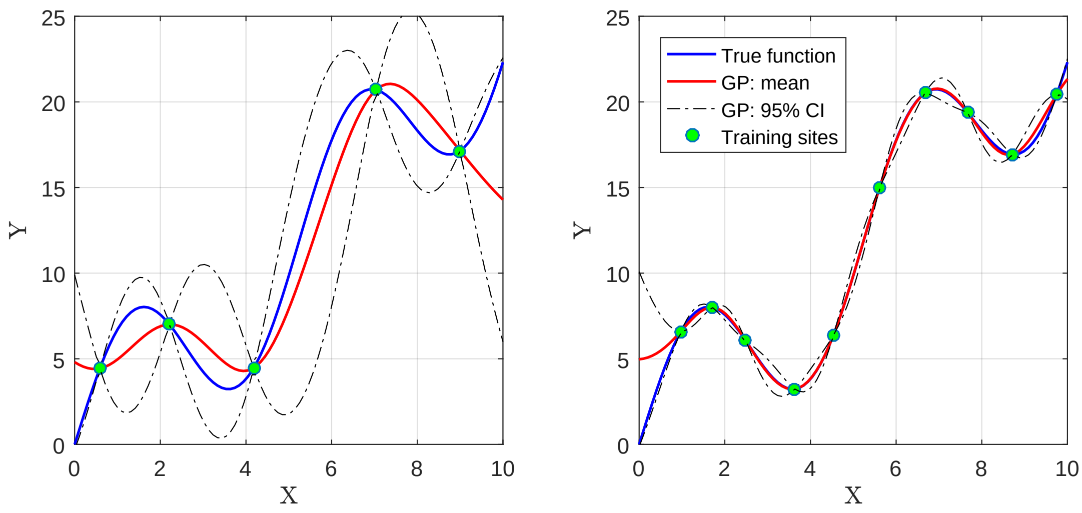


Fig. 4.2 A demonstration of the GP metamodeling with 5 training points (left) and 10 training points (right), based on a simple test function.

Figure 4.2 demonstrates GP metamodeling results for a simple test function over the range of $[0,10]$. The true function is only known at 5 and 10 training points respectively. The dash-dot lines correspond to 95% confidence intervals (CIs) of the prediction, which is the interval given by $\left[\mu_{\hat{y}}(\mathbf{x}^*) - 1.96 \cdot \sigma_{\hat{y}}^2(\mathbf{x}^*), \mu_{\hat{y}}(\mathbf{x}^*) + 1.96 \cdot \sigma_{\hat{y}}^2(\mathbf{x}^*)\right]$. From Figure 4.2 we can see that:

- GP metamodel always interpolates the design sites.
- The MSE of the prediction decreases as the untried point gets closer to training points. At training points the variance of prediction is zero. When the number of training sites increases from 5 to 10, the agreement of the GP mean with true function is greatly improved and the MSEs become very small.

Given the mathematical form of GP and the formulas for prediction and MSE, the process of using GP metamodeling involves a few other issues. They are: (1) estimation of hyperparameters, (2) selection of design/training sites, (3) assessment of metamodel accuracy to reproduce the computer model outputs.

4.2 Parameter Estimation

The formulas in Section 4.1 are presented given that we already know all the parameters in the GP metamodel. In this section the evaluation of the unknown parameters will be discussed, which is called the parameter estimation process. Right now we have $(2d+n+1)$ unknown parameters in the GP model:

1. n regression coefficients $\boldsymbol{\beta}$

2. 1 process variance $\{\sigma^2\}$[2]

3. d correlation kernel characteristic length-scales $\boldsymbol{\theta}$

4. d correlation kernel roughness parameters $\mathbf{p}$[3]

They are often referred to as **hyperparameters** of the GP emulator[4]: $\boldsymbol{\phi} = \left\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}, \mathbf{p}\right\}$. Parameter estimation is the process to find the set of parameters for the GP model that shows the best possible predictive performance. Parameter estimation is usually done by **Maximum Likelihood Estimation (MLE)** or **Cross Validation (CV)**. Detailed comparison of these two methods can be found in [6] [77] [86]. Both methods are based on solving an optimization problem as shown below.

4.2.1 Maximum Likelihood Estimation

Base on the fundamental assumption of GP metamodeling, the output values $\mathbf{y}$ of the design sites follow a joint Gaussian distribution:

$$\mathbf{y}|\boldsymbol{\phi} = \mathbf{y}|\left\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}, \mathbf{p}\right\} \sim \mathcal{N}\left(\mathbf{F}\boldsymbol{\beta}, \sigma^2\mathbf{R}\right) \tag{4.28}$$

The idea behind MLE is to find the set of hyperparameters $\boldsymbol{\phi}$ that maximize the likelihood of the observations $\mathbf{y}$. The likelihood function is defined as:

$$L\left(\mathbf{y}|\left\{\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}, \mathbf{p}\right\}\right) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^m \sqrt{\det(\mathbf{R})}} \exp\left[-\frac{(\mathbf{y}-\mathbf{F}\boldsymbol{\beta})^\top \mathbf{R}^{-1}(\mathbf{y}-\mathbf{F}\boldsymbol{\beta})}{2\sigma^2}\right] \tag{4.29}$$

where the correlation matrix $\mathbf{R}$ is dependent on $\{\boldsymbol{\theta}, \mathbf{p}\}$.

We first assume that the correlation kernel parameters $\{\boldsymbol{\theta}, \mathbf{p}\}$ are known, based on which we solve for $\left\{\boldsymbol{\beta}, \sigma^2\right\}$ in closed forms. The set of regression coefficients $\boldsymbol{\beta}$ are estimated by generalized least-squares as shown in Equation 4.13 and we copy it below. See Appendix D for the derivation.

$$\widehat{\boldsymbol{\beta}}|\{\boldsymbol{\theta}, \mathbf{p}\} = \widehat{\boldsymbol{\beta}}(\boldsymbol{\theta}, \mathbf{p}) = \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{y} \tag{4.30}$$

The log-likelihood function given the estimator $\widehat{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$ becomes:

[2]Note that some researchers often use the term "precision" defined as $\lambda = 1/\sigma^2$ [10] [61] [63] [80].

[3]Note that these parameters are usually known once we have chosen a certain correlation kernel. For example, for Gaussian kernels these parameters will all be 2.

[4]Some researchers only call correlation function-related parameters $\boldsymbol{\theta}$ and $\mathbf{p}$ as hyperparameters.

$$\begin{aligned}\log L\left(\mathbf{y}\middle|\left\{\widehat{\boldsymbol{\beta}},\sigma^2,\boldsymbol{\theta},\mathbf{p}\right\}\right) &= -\frac{m}{2}\log(2\pi)-\frac{1}{2}\log(\det(\mathbf{R}))\\ &\quad -m\log(\sigma)-\frac{\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)^\top\mathbf{R}^{-1}\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)}{2\sigma^2}\end{aligned}\tag{4.31}$$

The first two terms $\frac{m}{2}\log(2\pi)$ and $\frac{1}{2}\log(\det(\mathbf{R}))$ are constants because we have fixed $\{\boldsymbol{\theta},\mathbf{p}\}$. Because the log-function is monotically increasing, maximizing the log-likelihood is equivalent to maximizing the likelihood and minimizing the following temporary function:

$$g\left(\mathbf{y}\middle|\left\{\widehat{\boldsymbol{\beta}},\sigma^2,\boldsymbol{\theta},\mathbf{p}\right\}\right)=m\log(\sigma)+\frac{\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)^\top\mathbf{R}^{-1}\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)}{2\sigma^2}\tag{4.32}$$

Take the derivative with respect to σ:

$$\begin{aligned}\frac{\mathrm{d}g\left(\mathbf{y}\middle|\left\{\widehat{\boldsymbol{\beta}},\sigma^2,\boldsymbol{\theta},\mathbf{p}\right\}\right)}{\mathrm{d}\sigma} &= \frac{m}{\sigma}-\frac{\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)^\top\mathbf{R}^{-1}\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)}{\sigma^3}\\ &= \frac{m}{\sigma^3}\left[\sigma^2-\frac{\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)^\top\mathbf{R}^{-1}\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)}{m}\right]\end{aligned}\tag{4.33}$$

The estimator for minimizing $g\left(\mathbf{y}\middle|\left\{\widehat{\boldsymbol{\beta}},\sigma^2,\boldsymbol{\theta},\mathbf{p}\right\}\right)$ thus maximizing the log-likelihood function is given by:

$$\widehat{\sigma^2}|\{\boldsymbol{\theta},\mathbf{p}\}=\widehat{\sigma^2}(\boldsymbol{\theta},\mathbf{p})=\frac{1}{m}\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)^\top\mathbf{R}^{-1}\left(\mathbf{y}-\mathbf{F}\widehat{\boldsymbol{\beta}}\right)\tag{4.34}$$

Now we can substitute Equation 4.30 and Equation 4.34 into the likelihood function (Equation 4.29) or log-likelihood function (Equation 4.31), which will depend only on the correlation kernel parameters $\{\boldsymbol{\theta},\mathbf{p}\}$. If we choose the likelihood function, we get the so-called "**concentrated likelihood function**" [69]. Here we choose the log-likelihood function for numerical convenience:

$$\begin{aligned}-\log L\left(\mathbf{y}\middle|\left\{\widehat{\boldsymbol{\beta}},\widehat{\sigma^2},\boldsymbol{\theta},\mathbf{p}\right\}\right) &= \frac{m}{2}\log(2\pi)+\frac{1}{2}\log(\det(\mathbf{R}))+\frac{m}{2}\log\left(\widehat{\sigma^2}\right)+\frac{m}{2}\\ &= \frac{m}{2}\log\left(2\pi\widehat{\sigma^2}\right)+\frac{1}{2}\log(\det(\mathbf{R}))+\frac{m}{2}\end{aligned}\tag{4.35}$$

The MLE of $\{\boldsymbol{\theta},\mathbf{p}\}$ can be achieved by minimizing Equation 4.35:

$$\left\{\widehat{\boldsymbol{\theta}},\widehat{\mathbf{p}}\right\}=\arg\min_{\mathcal{D}(\boldsymbol{\theta},\mathbf{p})}\left(\frac{m}{2}\log\left(2\pi\widehat{\sigma^2}\right)+\frac{1}{2}\log(\det(\mathbf{R}))+\frac{m}{2}\right)\tag{4.36}$$

where $\mathcal{D}(\boldsymbol{\theta}, \mathbf{p})$ means the domain for possible values of $\{\boldsymbol{\theta}, \mathbf{p}\}$. Equation 4.30, 4.34 and 4.36 together are the MLE results of the hyperparameters. Note that this process can be made easier if we fix the values of **p** by choosing correlation kernels beforehand.

4.2.2 Cross Validation

The basic idea of CV is to leave out one or a few design sites and the corresponding observation(s) and then predict it (them) back based on the remaining points. The optimization problem is designed to minimize such prediction error which typically requires looping over all the design sites. Consider a general case of "***K*-fold CV**" [6] [77] which divides the domain $\mathcal{D}(\mathbf{X})$ of the design sites **X** into K mutually exclusive and collectively exhaustive subsets:

$$\mathcal{D}(\mathbf{X}) = \{\mathcal{D}_k, k = 1, 2, \dots, K\}$$

such that:

$$\mathcal{D}_i \cap \mathcal{D}_j = \varnothing \quad \forall (i, j) \in \{1, 2, \dots, K\}^2$$
$$\cup_{k=1}^{K} \mathcal{D}_k = \mathcal{D}(\mathbf{X})$$

Suppose we have left out the k^{th} set of design sites, we build the GP metamodel based on the remaining $(K-1)$ sets of observations and make predictions on that specific k^{th} set that we have removed. Such predictions are called "*cross-validated predictions*" and denoted by $\mu_{\hat{y},(-k)}\left(\mathbf{x}^{(k)}\right)$. A special case is when $K = m$ which means that every set only contains one design site. We repeat the process with $k = \{1, 2, \dots, K\}$ and then minimizing certain objective function to get estimation of the hyperparameters. A common choice of the objective function is given by [6] [121]:

$$\sum_{k=1}^{K} \left[y^{\mathrm{M}}\left(\mathbf{x}^{(k)}\right) - \mu_{\hat{y},(-k)}\left(\mathbf{x}^{(k)}\right) \right]^2 \tag{4.37}$$

The CV estimate of $\{\boldsymbol{\theta}, \mathbf{p}\}$ can be achieved by minimizing the above objective function:

$$\left\{\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{p}}\right\} = \arg \min_{\mathcal{D}(\boldsymbol{\theta}, \mathbf{p})} \sum_{k=1}^{K} \left[y^{\mathrm{M}}\left(\mathbf{x}^{(k)}\right) - \mu_{\hat{y},(-k)}\left(\mathbf{x}^{(k)}\right) \right]^2 \tag{4.38}$$

The CV estimate of σ^2 is calculated using [6] [25]:

$$\widehat{\sigma^2} \big| \left\{\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{p}}\right\} = \frac{1}{K} \sum_{k=1}^{K} \frac{\left[y^{\mathrm{M}}\left(\mathbf{x}^{(k)}\right) - \mu_{\hat{y},(-k)}\left(\mathbf{x}^{(k)}\right) \right]^2}{\sigma^2_{\hat{y},(-k)}\left(\mathbf{x}^{(k)}\right)} \tag{4.39}$$

where $\sigma^2_{\hat{y},(-k)}(\mathbf{x}^{(k)})$ is the MSE of the GP predictor based on all the training sites except for the k^{th} set $\mathcal{D}_k$. The most widely case of CV is when $K = m$ which means that every set only contains one design site. This is called the Leave-One-Out Cross-Validation (LOOCV).

4.2.3 Comparison of MLE and CV

Martin and Simpson [86] investigated six test problems to compare the performance of MLE and CV for parameter estimation. It was found that MLE was generally better than CV. CV has the potential to perform slightly better, but it also has the risk of performing much worse. In a recent study, Bachoc [6] also performed numerical study to compare MLE and CV and it was concluded that when the model is mis-specified, CV performs better than MLE. However, MLE is more likely to yield the best predictions as long as the correlation is properly specified. Therefore, CV is suitable for cases when one gives robustness over best possible performance.

Both MLE and CV result in multi-dimensional optimization problems as shown in above. Any local or global optimization techniques can be used but in many cases the optimal one is not known a priori. For more details see the listed local/global optimization techniques in [77] [85] [123] and several hybrid optimization approaches in [129]. During the optimization process for MLE, several issues exist, for example, the multi-modality and long ridges of the log-likelihood function and ill-conditioning of the correlation matrix. For more details and solutions the interested readers are referred to the discussion in [85] [86].

4.3 Experimental Design

A dense design matrix will produce a GP metamodel that has better predictive capability. However, this is in contradiction with the purpose of use GP, which is to greatly reduce the computational burden. A good balance can be achieved by improving the selection of the design sites, which is called "design of computer experiments" [119] [121]. Simple or crude Monte Carlo (MC) sampling [58] [60] selects samples randomly according to the specified probability distributions. Simple MC generally do not have a good coverage of the sampling space. Consequently, more samples are usually required to have similar accuracy than those space-filling designs such as Latin Hypercube Sampling (LHS) and low discrepancy sequences.

LHS [59] [92] is one of the stratified sampling technique, which divides the range of each input into N_{LHS} segments of equal probability. N_{LHS} is the number of desired samples. The relative lengths of the segments are determined by the nature of the prescribed PDF. For example, the segment lengths will be equal for a uniform distribution. LHS selects one sample randomly from each of the segments that have equal probability, resulting in N_{LHS} samples for each input. Based on a specified correlation structure, these samples are then combined in a shuffling operation to create a set of N_{LHS} multi-dimensional samples. Consequently, an obvious feature of resulting sample set is that every row and column has exactly one sample in the hypercube of partitions. LHS has been widely used for experimental designs. It requires fewer samples than simple MC sampling to achieve the same statistical accuracy, but it still can be prohibitively expensive. A popular variation of LHS that has been widely used in choosing training points is the "maximin LHS design" [68] [94], which select the one with the largest minimum distance among multiple designs to achieve a good coverage of the sampling space.

Low-discrepancy sequences, also called quasi-random or sub-random sequences [93] [100], is "less random" than a crude MC sequence. It contains samples that tend to be located "more uniformly" than crude MC samples. Here discrepancy refers to the non-uniformity of the samples within the unit hypercube. Low discrepancy sequences have an advantage over LHS design of being generated sequentially, which makes adding extra points convenient once required. They are widely used for approximation of integrals in higher dimensions or for global optimization because superior convergence can be achieved. Example of low discrepancy sequences are Sobol sequence [12] [128] and Halton sequence [93] [100].

By looking at the MSE we directly obtain a local error measure for any untried input location. This measure can be used to detect regions of low prediction accuracy, which can provide guidance to enrich the current experimental design. An overall increase in the GP metamodel prediction accuracy can be achieved by placing more samples to the region with large prediction variance. This procedure is referred to as "adaptive experimental design".

4.4 Accuracy Assessment

The GP metamodel's accuracy needs to be assessed before using it for further study. The quality of a metamodel is usually assessed with two approaches [86]: (1) accuracy to reproduce the design observations; (2) accuracy when predicting the computer model at untried locations. As GP is an interpolator it can reproduce the design samples exactly. Therefore, the first approach should be applied with CV. The second approach can be done by measuring the error when using GP to predict the QoIs for an independent "validation" or "test" samples. By "independent" it means that the validation sample set is different from the training sample set. However, this may require more simulator runs than those used for training the metamodels, which is in conflict with the motivation of using metamodels to reduce the computational cost.

CV and another method called Akaike's Information Criterion (AIC) were compared in [86]. Both CV and AIC do not require additional computer model runs to assess the accuracy of the metamodel. However, AIC is more appropriate for metamodel selection based on the predictive capability. Several approaches were presented in [9] for diagnostics of GP emulators, including individual prediction errors, Mahalanobis distance, variance decomposition and graphical methods, etc. Through literature review and our own experience, we found the following approaches are usually sufficient and most convenient to implement, including CV, predictivity coefficient and graphical inspection.

4.4.1 Cross Validation and Error Estimation

A CV procedure can be used to assess the accuracy of metamodels without sampling any additional points beyond those used to train the metamodels. In section 4.2 we have demonstrated that CV can be used for hyperparameter estimation. For validation of a metamodel, CV employs a similar idea to divide the training samples into K sets but only cares about the prediction error instead of minimizing the objective functions shown in Equation 4.38. Now we will use the LOOCV as an example as it is the most popular version of CV [86].

The basic idea is to leave out one observation and predict it back using the metamodel built based on the $(m-1)$ remaining points. By doing this in turn for each training sample we can get the residuals for each prediction, the average of which is called the “LOOCV error”:

$$\mathscr{E}_{\text{LOOCV}} = \frac{1}{m}\sum_{i=1}^{m}\left[y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y},(-i)}\left(\mathbf{x}^{(i)}\right)\right]^2 \tag{4.40}$$

In principle, evaluating the LOOCV error requires m different prediction residuals, which means we need to re-estimate the hyperparameters associated with m different GP metamodels. This is cumbersome and inconvenient for the validation. As suggested by Jones et al. [69], unless there are very few training samples or major outliers, dropping a single sample from the training set usually has a negligible effect on the MLE of hyperparameters. Therefore, in practice we keep the hyperparameters $\left\{\widehat{\sigma^2}, \widehat{\boldsymbol{\theta}}, \widehat{\mathbf{p}}\right\}$ evaluated based on all the training samples as constant, but only use the remaining $m-1$ points in computing regression coefficients $\widehat{\boldsymbol{\beta}}$, correlation matrix $\mathbf{R}$ and information matrix $\mathbf{F}$ used in Equation 4.12 - Equation 4.15.

4.4.2 Predictivity Coefficient

The **Predictivity Coefficient** Q_2 gives the percentage of the output variance explained by the emulator:

$$Q_2 = 1 - \frac{\sum_{i=1}^{N_{\text{val}}}\left[y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y}}\left(\mathbf{x}^{(i)}\right)\right]^2}{\sum_{i=1}^{N_{\text{val}}}\left[y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \overline{y^{\text{M}}}\right]^2} \tag{4.41}$$

Where N_{val} is the size of the validation sample set, $y^{\text{M}}(\mathbf{x}^{(i)})$ is the output from full model simulation, $\overline{y^{\text{M}}}$ is their empirical mean value and $\mu_{\hat{y}}(\mathbf{x}^{(i)})$ is the predicted value using GP metamodel. A Q_2 value close to 1.0 means that the model is accurate. In practical situations, a metamodel with Q_2 value above 0.7 is often considered as a satisfactory approximation of the full model [66] [84]. However this requires full model executions at perhaps a larger sample set than that used to train the GP model.

Equation 4.41 can be re-wrote in a form that incorporates CV errors. In this case, no further simulator runs are required. Note that N_{val} becomes the size of training sample set m. The "cross-validated predictions" $\mu_{\hat{y},(-i)}(\mathbf{x}^{(i)})$ are also used to replace $\mu_{\hat{y}}(\mathbf{x}^{(i)})$.

$$Q_2 = 1 - \frac{\sum_{i=1}^{m} \left[y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y},(-i)}\left(\mathbf{x}^{(i)}\right) \right]^2}{\sum_{i=1}^{m} \left[y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \overline{y^{\text{M}}} \right]^2} \tag{4.42}$$

4.4.3 Graphical Inspection

The difference between simulator and emulator runs for error estimation. Define the following:

$$\mathscr{E}_{\text{IPE}}\left(\mathbf{x}^{(i)}\right) = y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y}}\left(\mathbf{x}^{(i)}\right) \tag{4.43}$$

$$\mathscr{E}_{\text{IPE, CV}}\left(\mathbf{x}^{(i)}\right) = y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y},(-i)}\left(\mathbf{x}^{(i)}\right) \tag{4.44}$$

$$\mathscr{E}_{\text{ISE}}\left(\mathbf{x}^{(i)}\right) = \frac{y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y}}\left(\mathbf{x}^{(i)}\right)}{\sqrt{\sigma_{\hat{y}}^2\left(\mathbf{x}^{(i)}\right)}} \tag{4.45}$$

$$\mathscr{E}_{\text{ISE, CV}}\left(\mathbf{x}^{(i)}\right) = \frac{y^{\text{M}}\left(\mathbf{x}^{(i)}\right) - \mu_{\hat{y},(-i)}\left(\mathbf{x}^{(i)}\right)}{\sqrt{\sigma_{\hat{y},(-i)}^2\left(\mathbf{x}^{(i)}\right)}} \tag{4.46}$$

They are called "individual prediction errors (IPE)", "CV individual prediction errors", "individual standardized errors (ISE)" and "CV individual standardized errors", respectively. Note that the term "errors" in these definitions are sometimes called "residuals" in previous literature [69]. The definitions of $\mathscr{E}_{\text{IPE}}$ and $\mathscr{E}_{\text{ISE}}$ can be found in some earlier work [9], while $\mathscr{E}_{\text{IPE, CV}}$ and $\mathscr{E}_{\text{ISE, CV}}$ are defined in the current work to avoid additional simulator runs.

Graphical inspection is an efficient way to investigate the quality of GP metamodel and to verify the Gaussian assumption when building the emulator. The following plots can be made and inspected. Note that a GP metamodel of high quality should pass the diagnostic criteria of all the following graphs.

- Plot the predictions from the simulator against the emulator. For an accurate metamodel, the points on this graph should fall on the diagonal line.

- Plot $\mathscr{E}_{\mathrm{IPE}}$ or $\mathscr{E}_{\mathrm{IPE,\,CV}}$ against the emulator's predictions. This graph is useful for detecting problems in the mean function of the GP metamodel. Patterns like systematic positive or negative errors in some particular ranges of the outputs indicate mis-specifications of the mean function.

- Plot $\mathscr{E}_{\mathrm{ISE}}$ or $\mathscr{E}_{\mathrm{ISE,\,CV}}$ against the emulator's predictions. Because $\mathscr{E}_{\mathrm{ISE}}$ or $\mathscr{E}_{\mathrm{ISE,\,CV}}$ are $\mathscr{E}_{\mathrm{IPE}}$ or $\mathscr{E}_{\mathrm{IPE,\,CV}}$ divided by the standard deviations of the emulator's prediction variances, this graph also provide some information for the prediction variances. Firstly, if the underlying Gaussian assumption is valid, 99.7% of the points should lie within the interval $[-3,3]$ [69]. Secondly, these standardized errors should not demonstrate systematic bias.

- Quantile-Quantile plot, which is the quantiles of $\mathscr{E}_{\mathrm{ISE}}$ or $\mathscr{E}_{\mathrm{ISE,\,CV}}$ versus the quantiles from random samples of the same size from a standard normal distribution. The points should lie close to the 45° line, suggesting that the standardized residuals are close to being normally distributed.

4.4.4 CV vs. generating new test samples

It is generally up to the user to select between using CV or generating new test samples for evaluating the accuracy of the GP metamodel. Firstly, the computational cost should be considered. Emulator is chosen because the inverse UQ involves hundreds of thousands of simulator runs. But if a few hundred extra runs are easily affordable, we can just generate new test runs for metamodel validation, this is referred to as "test sample approach" in literature [66]. Secondly, it has to be noted that test sample approach can provide misleading and erroneous results, especially when the test sample size is small and the test sample locations are not properly chosen. For example, if most of the limited number of test samples are generated close to training samples, the GP metamodel will nearly "interpolate" the test samples, giving the wrong impression that the metamodel is accurate. A "sequential validation design" was proposed in [66] that can put test sample points in the unfilled region of the training sample design. This algorithm can optimize the distance between the test set and training set and assess the metamodel predictivity with a minimum number of test sample points.

4.5 Implementation Issues

Many packages or codes have been developed to implement GP metamodeling, examples are DACE [81], DAKOTA [1] [2], DiceKriging [118], GPML [114], GPM/SA [42] [145] and UQLab [83]. We would like to mention some techniques which identify the mean functions through some data-analytic procedures, for example, Blind Kriging [70] and Polynomial-chaos-based Kriging [123]. However, one should bear in mind that such processes will add the complexity of GP model and thus the computational cost, which may eventually outweigh the increased accuracy. Some other variations of Kriging take advantage of the derivative information, for example, the gradient-enhanced Kriging and Hessian-enhanced Kriging [2] [38]. These techniques clearly have the potential of building more accurate predictions, at the cost of increased size of the correlation matrix and more expensive parameter estimation.

In the parameter estimation process using MLE, all calculations should be performed in logarithms to avoid issues with finite precision arithmetic. The correlation matrix $\mathbf{R}$ needs to be inverted at various stages, e.g., evaluation of the regression coefficients by Equation 4.13, the predictor by Equation 4.12, the MSE by Equation 4.14 or 4.15. The size of the correlation matrix is $(m \times m)$ which increases with the number of design sites. Inversion of such a matrix is well known to suffer from numerical instabilities, especially when two design sites are close, resulting in very similar columns. To solve this problem, a widely used practice is to add a small value τ^2 to the diagonal entries of $\mathbf{R}$, which is called the "nugget" or "jitter" $\mathbf{R}_{ii} = 1 + \tau^2$. Such a term serves as a noise factor and will prevent the interpolation of the training sites. It is a convenient way to make sure the covariance matrix is always invertible by introducing negligible errors.

When building the GP emulator, the training inputs are suggested to be scaled between 0 and 1, which correspond to minimum and maximum values from the training set, respectively. Furthermore, the output data can be centralized and standardized so that they have mean 0 and variance 1. Such data processing can reduce the arithmetic errors in matrix inversion and is suggested in [4] [61] [63] [86].

Finally, like almost any other computational tools, an important question for GP is how does the computational cost increase with the dimensionality. Intuitively higher dimension causes more space between points and the number of training/design points will increase rapidly with the number of inputs. However, in practice computer models never respond strongly to all of their input parameters. Adaptive construction of the GP emulator is possible by progressively assigning more training points to important dimensions which has less smoothness. It was claimed in [104] that GP can be implemented effectively with up to 50 inputs. Considering the booming computational power in the last decade, GP can deal with much higher dimensions with modern computing platforms.

Chapter 5

APPLICATION TO SIMPLIFIED REACTOR SIMULATIONS

In this chapter, inverse UQ is applied to a simplified nuclear reactor simulation problem, the Point Reactor Kinetics Equation (PRKE) coupled with lumped parameter TH feedback model. Metamodel constructed by PCE is adopted during MCMC sampling and direct numerical simulation is also performed as a reference solution. In this case, MCMC sampling using the metamodel will only be evaluation of polynomials which has negligible computational cost. The work presented in this chapter is also published in [149] [150] [159].

5.1 Problem Definition

In this Section, the simplified reactor simulation model used to demonstrate the inverse UQ under the Bayesian framework is introduced, which is the PRKE coupled with lumped parameter TH feedback model [64] [112]. Similar model has also been used in [51] to demonstrate forward UQ capability of PCE. In the current research, we have elaborated this model to make it more representative of a real Pressurized Water Reactor (PWR).

5.1.1 Simplified Reactor Simulation Model

The PRKE with lumped parameter TH feedback model describes the transient behavior of the normalized power level $p(t)$, the delayed neutron precursor concentrations $C(t)$, and the core effective fuel and coolant temperatures, $T_{\text{fuel}}(t)$ and $T_{\text{cool}}(t)$. The model employs the standard neutron point kinetic equations and couples them to simple 0-D (core average) fuel heat conduction and fluid energy balance models via the reactivity function. In this model, the reactivity depends on both fuel and coolant temperatures. The system of coupled nonlinear ODEs is given by:

$$\frac{dp(t)}{dt} = \frac{\rho(t, T_{\text{fuel}}, T_{\text{cool}}) - \beta}{\Lambda} p(t) + \lambda C(t) \tag{5.1}$$

$$\frac{dC(t)}{dt} = \frac{\beta}{\Lambda} p(t) - \lambda C(t) \tag{5.2}$$

$$\frac{dT_{\text{fuel}}(t)}{dt} = \frac{\Omega_{\text{pow}}}{\rho_{\text{fuel}} c_{p,\text{fuel}}} p(t) - \frac{1}{\rho_{\text{fuel}} c_{p,\text{fuel}} \hat{R}_{th}} \left[T_{\text{fuel}}(t) - T_{\text{cool}}(t)\right] \tag{5.3}$$

$$\frac{dT_{\text{cool}}(t)}{dt} = -\frac{2u}{H}\left[T_{\text{cool}} - T_{\text{cool}}^{\text{in}}\right] + \frac{A_{\text{fuel}}}{A_{\text{flow}}} \frac{1}{\rho_{\text{cool}} c_{p,\text{cool}} \hat{R}_{th}} \left[T_{\text{fuel}}(t) - T_{\text{cool}}(t)\right] \tag{5.4}$$

Note that only one-group effective delayed neutron is considered. Table 5.1 shows the parameters definition and values used in the model.

The fuel and cladding material properties are based on LWR fuel and cladding properties [154] [157]. The lumped parameter (i.e. averaging the unknown values over the whole domain) description of the reactor fuel and coolant temperatures allows the elimination of the spatial dependencies and therefore focuses on the time-dependent part. In the TH equations, the thermal resistance $\hat{R}_{th}$ accounts for the conduction through the fuel pellet, gap and cladding, and the convection at the clad-coolant interface. The lumped fluid equation is based on an enthalpy conservation for a fluid being advected and heated in a channel of hydraulic cross sectional area A_{flow}. The convection heat transfer coefficient h_{conv} is given by the Dittus-Boelter correlation for turbulent-flow convection.

$$\hat{R}_{th} = \frac{A_{\text{fuel}}}{2\pi} \cdot \left[\frac{1}{R_{\text{gap}} h_{\text{gap}}} + \frac{1}{k_{\text{clad}}} \ln \frac{R_{\text{fuel}}}{R_{\text{gap}}} + \frac{1}{R_{\text{clad}} h_{\text{conv}}(T_{\text{cool}})} + \frac{w}{2k_{\text{fuel}}(T_{\text{fuel}})}\right] \tag{5.5}$$

$$\frac{h_{\text{conv}}(T_{\text{cool}}) \cdot L}{k_{\text{cool}}(T_{\text{cool}})} = 0.023 \times \left[\frac{\rho_{\text{cool}}(T_{\text{cool}}) u L}{\mu_{\text{cool}}(T_{\text{cool}})}\right]^{0.8} \times \left[\frac{c_{p,\text{cool}}(T_{\text{cool}}) \mu_{\text{cool}}(T_{\text{cool}})}{k_{\text{cool}}(T_{\text{cool}})}\right]^{0.4} \tag{5.6}$$

The parameters used to calculate $\hat{R}_{th}$ are shown in Table 5.2. Coolant properties $\rho_{\text{cool}}(T_{\text{cool}})$, $c_{p,\text{cool}}(T_{\text{cool}})$, $\mu_{\text{cool}}(T_{\text{cool}})$ and $k_{\text{cool}}(T_{\text{cool}})$ all depend on coolant temperature at a given constant pressure. These properties are calculated using MATLAB water properties functions **Xsteam**.

In this model, the neutronics equations (the first two equations) are coupled to the TH equations (i.e., the last two equations) via the temperature dependent total reactivity $\rho(t, T_{\text{fuel}}, T_{\text{cool}})$ that contains the external reactivity term (e.g., control rod movement), the fuel temperature

Table 5.1 Parameters used in the PRKE coupled with lumped parameter TH feedback model

Variable	Detail	Unit	Value
$p(t)$	normalized nuclear reactor power		$p(0) = 1$
$C(t)$	normalized precursor concentration		
$\rho(t, T_{\text{fuel}}, T_{\text{cool}})$	total reactivity		
$T_{\text{fuel}}(t)$	average fuel temperature	K	
$T_{\text{cool}}(t)$	average coolant temperature	K	
λ	decay constant of the precursor	s^{-1}	7.662e-02
β	total delayed neutron fraction		6.500e-03
Λ	neutron mean generation time	s	5.596e-05
$\rho_{\text{fuel}}(T_{\text{fuel}})$	fuel density	kg/m^3	
$\rho_{\text{cool}}(T_{\text{cool}})$	coolant density	kg/m^3	**Xsteam**
$c_{p,\text{fuel}}(T_{\text{fuel}})$	fuel specific heat	$J/(kg \cdot K)$	
$c_{p,\text{cool}}(T_{\text{cool}})$	coolant specific heat	$J/(kg \cdot K)$	**Xsteam**
$\Omega_{\text{pow}} = P_0/V_{\text{fuel}}$	conversion factor from normalized power $p(t)$ into power densities	$W/(m^3)$	3.530e+08
P_0	total initial power in the entire core	W	
V_{fuel}	fuel volume in the entire core	m^3	
$A_{\text{fuel}} = \pi R_{\text{fuel}}^2$	surface of fuel pin of radius R_{fuel}	m^2	5.281e-05
A_{flow}	average flow area around a fuel pin	m^2	8.995e-05
$\hat{R}_{th}$	fuel thermal resistance	$(m^3 \cdot K)/W$	
u	inlet coolant flow velocity	m/s	5.0
H	reactor height	m	4.0
$T_{\text{cool}}^{\text{in}}$	coolant inlet temperature	K	563.15

reactivity and the coolant temperature reactivity:

$$\rho(t, T_{\text{fuel}}, T_{\text{cool}}) = \rho_{\text{ext}} - \alpha_D[T_{\text{fuel}}(t) - T_{\text{fuel}}(0)] - \alpha_c[T_{\text{cool}}(t) - T_{\text{cool}}(0)] \tag{5.7}$$

In the above equation, ρ_{ext}, α_D and α_c are external reactivity insertion, Doppler reactivity coefficient and coolant temperature coefficient, respectively. These three parameters are treated as uncertain input parameters in the model. The QoIs in this model are $p(t)$, $C(t)$, $T_{\text{fuel}}(t)$ and $T_{\text{cool}}(t)$.

Table 5.2 Parameters used to calculate $\hat{R}_{th}$

Variable	Detail	Unit	Value
R_{fuel}	single fuel pin radius	m	0.0041
R_{gap}	gap radius	m	0.00411
R_{clad}	clad radius	m	0.00468
h_{gap}	gap conductance	$W/(m^2 \cdot K)$	1.0e+04
$h_{\text{conv}}(T_{\text{cool}})$	wall-coolant forced convection heat exchange coefficient	$W/(m^2 \cdot K)$	**Xsteam**
L	characteristic length	m	0.02447
$k_{\text{fuel}}(T_{\text{fuel}})$	fuel thermal conductivity	$W/(m \cdot K)$	
$k_{\text{clad}}(T_{\text{fuel}})$	clad thermal conductivity	$W/(m \cdot K)$	
w	weighting factor used in the effective fuel temperature formula: $T_{\text{fuel}} = wT_{\text{fuel}}^{\text{centerline}} + (1-w)T_{\text{fuel}}^{\text{surface}}$		4/9
$k_{\text{cool}}(T_{\text{cool}})$	coolant thermal conductivity	$W/(m \cdot K)$	**Xsteam**
$\mu_{\text{cool}}(T_{\text{cool}})$	coolant dynamic viscosity	$\text{Pa} \cdot s$	**Xsteam**
P	coolant pressure	MPa	15.5

5.1.2 Uncertain Input Parameters

For forward UQ with both MC sampling and PCE, prior uncertainties are required. The three random input parameters are modelled as lognormal random variables because we want them to be strictly positive (even though in reality they do not have to be positive). Normal distributions are avoided here because all of the three input parameters have very small mean values, and they can easily fall into the negative part if modelled as Gaussian. By this lognormal definition, we will be able to see a transient in which the power will increase very fast during a short time period and then the system will go to a new steady state because of the negative reactivity feedback from the fuel and coolant.

Table 5.3 Priors of three random input parameters “std” represents standard deviation)

Parameters	Prior 1		Prior 2		Prior 3	
	mean	std/mean	mean	std/mean	mean	std/mean
ρ_{ext}	$0.8 \cdot \rho_{\text{ext},0}$	20%	$0.8 \cdot \rho_{\text{ext},0}$	10%	$1.1 \cdot \rho_{\text{ext},0}$	20%
α_D	$0.8 \cdot \alpha_{D,0}$	40%	$0.8 \cdot \alpha_{D,0}$	20%	$1.5 \cdot \alpha_{D,0}$	40%
α_c	$0.8 \cdot \alpha_{c,0}$	40%	$0.8 \cdot \alpha_{c,0}$	20%	$1.5 \cdot \alpha_{c,0}$	40%

Three different priors are considered here and they are shown in Table 5.3. They will be referred to as Prior 1, Prior 2 and Prior 3. In Table 5.3, $\rho_{\text{ext},0} = \beta$, $\alpha_{D,0} = 3.18 \times 10^{-3}\beta$, $\alpha_{c,0} = 8.2186 \times 10^{-3}\beta$ are reference values used in [64]. Prior 1 and Prior 2 have the same mean values but different variances, while Prior 3 has different mean values. Figure 5.1 shows the comparison of PDFs based on the priors.

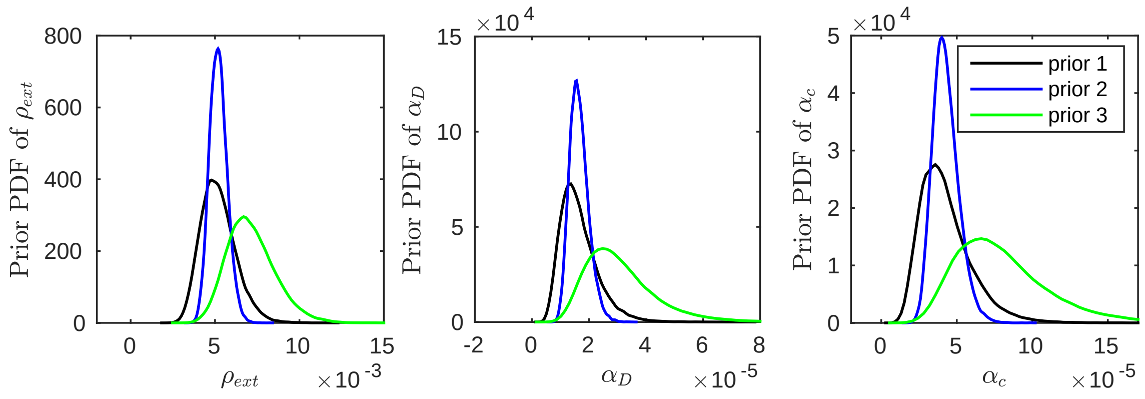


Fig. 5.1 Prior PDFs of three lognormal distributed random input parameters.

5.2 Formulation of Stochastic Problem by PCE

5.2.1 Approximate Uncertain Input Parameters with PCE

To construct the stochastic version of the model using PCE, first the random input quantities are approximated using the PCE in the following way:

$$\rho_{\text{ext}}(\xi_1) = \sum_{i=0}^{P} \rho_{\text{ext}}^{i} \Psi_i(\xi_1) \tag{5.8}$$

$$\alpha_{\text{D}}(\xi_2) = \sum_{i=0}^{P} \alpha_{\text{D}}^{i} \Psi_i(\xi_2) \tag{5.9}$$

$$\alpha_{\text{c}}(\xi_3) = \sum_{i=0}^{P} \alpha_{\text{c}}^{i} \Psi_i(\xi_3) \tag{5.10}$$

Hermite orthogonal polynomials are chosen in this problem. The reason is that lognormal PDFs are close to normal PDFs in shape, so we expect a relatively low order expansion in Hermite polynomials to be accurate. Note that right now we are ignorant of the distributions of the four QoIs. We can still use the Hermite polynomials and if QoIs are not Gaussian we will obtain a slower convergence.

The question now is how to approximate lognormal random variables with Hermite polynomials? We start with a general case in which k is an arbitrary type random variable that we want to approximate with orthogonal polynomials $\{\Psi_i(\xi)\}_{i=0}^{P}$, where ξ is another random variable.

$$k = \sum_{i=0}^{P} k_i \Psi_i(\xi) \tag{5.11}$$

To solve for the expansion coefficients $\{k_i\}_{i=0}^{P}$ we start with the spectral projection.

$$k_i = \frac{\langle k, \Psi_i \rangle}{\langle \Psi_i^2 \rangle} = \frac{1}{\langle \Psi_i^2 \rangle} \int k \Psi_i(\xi) g(\xi) d\xi, \quad i = 0, 1, 2, \ldots \tag{5.12}$$

where $g(\xi)$ is the weighting function for the chosen orthogonal polynomials. In order to conduct the above projection, we need to transform the full correlated random variable k and ξ to the same probability space. Recall that the inverse CDF of any distribution follows the uniform distributions $u \in \mathcal{U}(0,1)$. Define the PDFs for k and ξ as $f(k)$ and $g(\xi)$ respectively:

$$du = f(k)dk = dF(k)$$
$$du = g(\xi)d\xi = dG(\xi)$$

where $F(k)$ and $G(\xi)$ are the CDFs for k and ξ respectively:

$$F(k) = \int_{-\infty}^{k} f(t)dt$$
$$G(\xi) = \int_{-\infty}^{\xi} g(t)dt$$

Since we are transferring k and ξ to the same uniform random variable $u \in \mathcal{U}(0,1)$:

$$k = F^{-1}(u)$$
$$\xi = G^{-1}(u)$$
$$k_i = \frac{1}{\langle \Psi_i^2 \rangle} \int k \Psi_i(\xi) g(\xi) d\xi = \frac{1}{\langle \Psi_i^2 \rangle} \int_0^1 F^{-1}(u) \Psi_i\left(G^{-1}(u)\right) du \tag{5.13}$$

Once we have chosen the orthogonal polynomials $\{\Psi_i(\xi)\}_{i=0}^{P}$, $\langle \Psi_i^2 \rangle$ will be known by simple numerical integration. To evaluate the above integral, we can use MC sampling or Gaussian quadrature rules easily.

Tables 5.4 - 5.6 show the PCE modes for the three lognormal input parameters using Hermite polynomials, for different priors and up to order 10 (for one variable, number of PCE modes equals polynomial order plus one). Data in Table 5.4 - 5.6 are visualized in Figure 5.2. It can be seen that by increasing polynomial order, the expansion modes decay quickly to zero, indicating that we do not need such high order PCE to approximate these lognormal input parameters.

Table 5.4 PCE modes for three lognormal input parameters, Prior 1

Modes index	ρ_{ext}	α_{D}	α_{c}
0	5.20E-03	1.65E-05	4.27E-05
1	1.03E-03	6.37E-06	1.65E-05
2	1.02E-04	1.23E-06	3.17E-06
3	6.51E-06	1.56E-07	4.02E-07
4	-1.77E-08	1.25E-08	3.23E-08
5	-2.50E-07	-1.19E-09	-3.09E-09
6	-2.36E-07	-1.61E-09	-4.15E-09
7	-9.66E-08	-8.41E-10	-2.17E-09
8	-5.03E-08	-3.43E-10	-8.86E-10
9	-1.01E-08	-8.69E-11	-2.25E-10
10	-1.57E-09	-1.04E-11	-2.70E-11

Table 5.5 PCE modes for three lognormal input parameters, Prior 2

Modes index	ρ_{ext}	α_{D}	α_{c}
0	5.20E-03	1.65E-05	4.27E-05
1	5.19E-04	3.27E-06	8.46E-06
2	2.58E-05	3.24E-07	8.37E-07
3	7.63E-07	2.07E-08	5.35E-08
4	-2.27E-07	-5.62E-11	-1.45E-10
5	-1.18E-07	-7.94E-10	-2.05E-09
6	-1.74E-07	-7.52E-10	-1.94E-09
7	-4.39E-08	-3.07E-10	-7.94E-10
8	-3.78E-08	-1.60E-10	-4.13E-10
9	-4.61E-09	-3.21E-11	-8.29E-11
10	-1.21E-09	-5.01E-12	-1.29E-11

Figure 5.3 shows the convergence of approximation of the prior mean values and standard deviations by Hermite polynomials up to PCE of order 5. Y-axis is the absolute difference between PCE approximation and true values shown in Table 5.3. It can be seen that increasing the PCE order does not improve the mean value since the error is already extremely small with order 1. The approximation error of standard deviations stops decreasing at order 3. In the following analysis, PCE of order 3 will be used to solve for the expansion modes for the three random input parameters.

Table 5.6 PCE modes for three lognormal input parameters, Prior 3

Modes index	ρ_{ext}	α_{D}	α_{c}
0	7.15E-03	3.10E-05	8.01E-05
1	1.42E-03	1.19E-05	3.09E-05
2	1.40E-04	2.30E-06	5.94E-06
3	8.95E-06	2.92E-07	7.54E-07
4	-2.43E-08	2.35E-08	6.06E-08
5	-3.43E-07	-2.24E-09	-5.79E-09
6	-3.25E-07	-3.01E-09	-7.78E-09
7	-1.33E-07	-1.58E-09	-4.07E-09
8	-6.91E-08	-6.43E-10	-1.66E-09
9	-1.39E-08	-1.63E-10	-4.21E-10
10	-2.16E-09	-1.96E-11	-5.06E-11

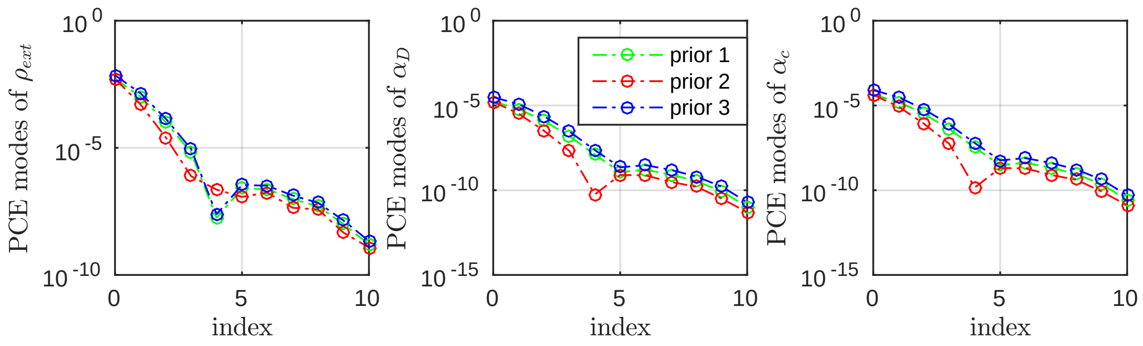


Fig. 5.2 PCE modes for approximation of uncertain input parameters with Hermite polynomials

5.2.2 Derivation of the Stochastic Version of the Model

Once we have the PCE modes for the random input parameters, we can perform similar expansion for the four QoIs, with $\boldsymbol{\xi} = [\xi_1, \xi_2, \xi_3]^\top$:

$$p(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} p^i(t)\Psi_i(\boldsymbol{\xi}) \tag{5.14}$$

$$C(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} C^i(t)\Psi_i(\boldsymbol{\xi}) \tag{5.15}$$

$$T_{\text{fuel}}(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} T^i_{\text{fuel}}(t)\Psi_i(\boldsymbol{\xi}) \tag{5.16}$$

$$T_{\text{cool}}(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} T^i_{\text{cool}}(t)\Psi_i(\boldsymbol{\xi}) \tag{5.17}$$

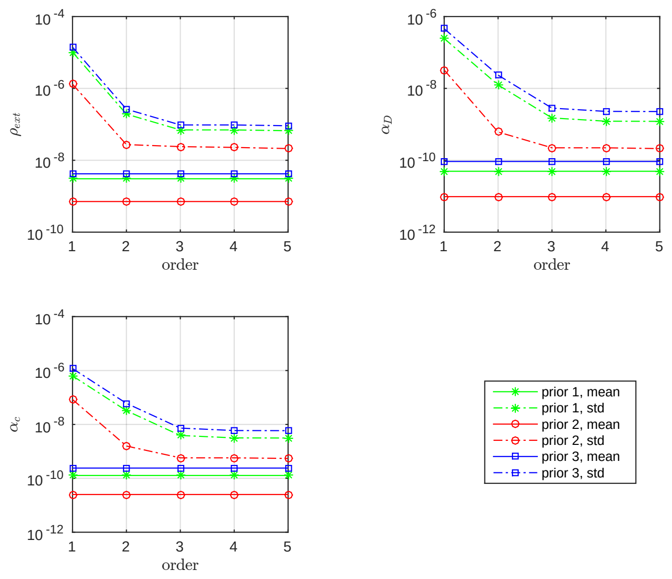


Fig. 5.3 Convergence of the approximations of prior mean values and standard deviations by PCE up to polynomial order 5

Note that in the previous subsection three input parameters (ρ_{ext}, α_{D} and α_{c}) are expanded with three i.i.d. standard Gaussian random variables (ξ_1, ξ_2, ξ_3), respectively, because they are mutually independent. However, when solving for the expansion of the four QoIs, they should depend on all of the three standard Gaussian random variables. That's why $\boldsymbol{\xi} = [\xi_1, \xi_2, \xi_3]^\top$ is used.

Substituting the above expansions into the PRKE coupled system of ODEs and projecting the four equations on the s^{th} Hermite polynomial for $s = 0, 1, ..., P$, we obtain:

$$\frac{dp^s(t)}{dt} = -\frac{\beta}{\Lambda}p^s(t) + \lambda C^s(t) + \frac{1}{\Lambda}\sum_{i=0}^{P}\sum_{j=0}^{P}\mathbf{T}_{ijs}\left[\rho_{\text{ext}}^i + T_{\text{fuel}}(0)\alpha_{\text{D}}^i + T_{\text{cool}}(0)\alpha_{\text{c}}^i\right]p^j(t) \\ -\frac{1}{\Lambda}\sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{l=0}^{P}\mathbf{T}_{ijls}\left[\alpha_{\text{D}}^i T_{\text{fuel}}^l(t) + \alpha_{\text{c}}^i T_{\text{cool}}^l(t)\right]p^j(t) \quad (5.18)$$

$$\frac{dC^s(t)}{dt} = \frac{\beta}{\Lambda}p^s(t) - \lambda C^s(t) \quad (5.19)$$

$$\frac{dT_{\text{fuel}}^s(t)}{dt} = \frac{\Omega_{\text{power}}}{\rho_{\text{fuel}}c_{p,\text{fuel}}}p^s(t) - \frac{1}{\rho_{\text{fuel}}c_{p,\text{fuel}}\hat{R}_{\text{th}}}\left[T_{\text{fuel}}^s(t) - T_{\text{cool}}^s(t)\right] \quad (5.20)$$

$$\frac{dT_{\text{cool}}^s(t)}{dt} = \frac{A_{\text{fuel}}}{A_{\text{cool}}}\frac{1}{\rho_{\text{cool}}c_{p,\text{cool}}\hat{R}_{\text{th}}}\left[T_{\text{fuel}}^s(t) - T_{\text{cool}}^s(t)\right] - \frac{2u}{H}\left[T_{\text{cool}}^s(t) - T_{\text{cool}}^{\text{in}}\cdot\delta_{0s}\right] \quad (5.21)$$

The derivation is presented in Appendix B. The new system of ODEs is P times larger than the original system of ODEs. They can be solved by classical Runge-Kutta methods. It has to be mentioned that the model analyzed in the current research is not computationally prohibitive (each simulation takes less than one second) and it is only used to demonstrate the applicability of PCE as a metamodel. We are able generate up to 50,000 MCMC samples in an acceptable time period using direct model simulation, which serves as a reference solution to those using PCE surrogates.

5.3 Forward UQ Results

In this part, forward UQ results are presented. The simulation results for the four QoIs are first shown at the nominal values of three random input parameters to demonstrate the transient evolution of the model. Then the mean values and standard deviations from MC sampling and PCE are compared.

5.3.1 Simulation Results at Nominal Values of Uncertain Input Parameters

We assume that the system starts from the following initial conditions:

$$\begin{aligned}
p(t=0) &= p_0 = 1.0 \\
C(t=0) &= \frac{\beta}{\lambda\Lambda} p_0 \\
T_{\text{fuel}}(t=0) &= T_{\text{cool}}(0) + \Omega_{\text{pow}} \cdot \hat{R}_{th} \cdot p_0 \\
T_{\text{cool}}(t=0) &= T_{\text{cool}}^{\text{in}} + \frac{H}{2u} \cdot \frac{A_{\text{fuel}}}{A_{\text{flow}}} \cdot \frac{\Omega_{\text{pow}} p_0}{\rho_{\text{cool}} c_{p,\text{cool}}}
\end{aligned}$$

Figures 5.4 and Figure 5.5 show the simulation results at nominal values of the three uncertain input parameters. Prior 1 and Prior 2 have the same results as the mean values are the same. With the introduction of external reactivity, the power quickly increases to about 4 and 10 times of the initial state. Consequently, the fuel and coolant temperature increase introduces negative feedback to counter-interact the external reactivity, and the power returns to a new lower value. The increase in coolant temperature is small and the negative reactivity feedback caused by the coolant is negligible, which causes the system to be insensitive to the coolant temperature coefficient α_c.

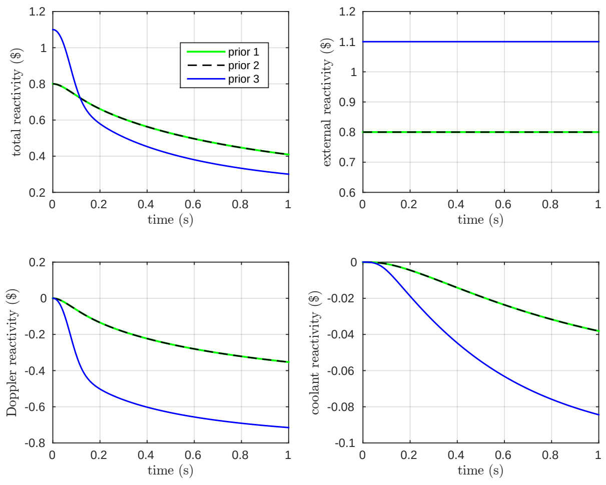


Fig. 5.4 Reactivity evolutions at nominal values of the uncertain input parameters

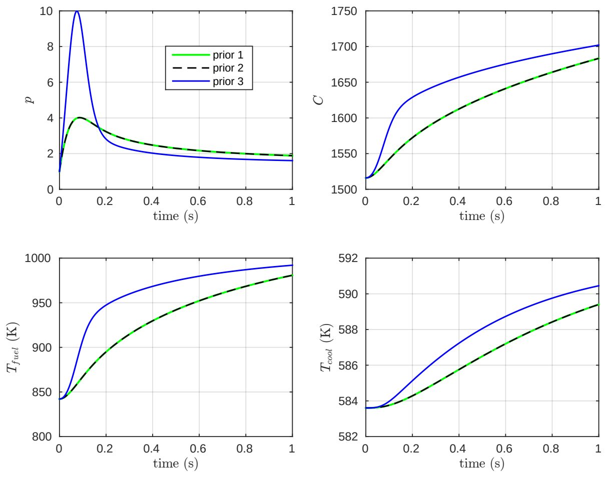


Fig. 5.5 Simulation results of four QoIs at nominal values of the uncertain input parameters

5.3.2 Forward UQ Results with PCE

In this part the superiority of PCE for forward UQ is shown. Given the prior uncertainties for three random inputs, we perform PCE with intrusive Galerkin projection approach up to order 5 to solve for the modes. Table 5.7 shows the number of PCE terms required with order 1 to 5, for the current problem with three uncertain input parameters.

Table 5.7 Number of terms required by PCE of order 1 to 5

PCE order	1	2	3	4	5
Number of terms	4	10	20	35	56

Figure 5.6 shows the convergence of mean values and standard deviations of four QoIs by MC sampling at the peak power time ($\sim$0.1s) using Prior 1 (Prior 2 and Prior 3 have similar results). Y axis is the absolute change in mean and standard deviation when more samples are added. 5000 samples are sufficient to produce a good reference solution to mean and standard deviations as the absolute changes are already very small. The large fluctuations in Figure 8 are caused by the high sensitivity of the model to input parameters, especially ρ_{ext}. As shown in Figure 5.5, changing from Prior 1 to Prior 3 causes the peak power to increase from 4 times to 10 times of the initial value. If a certain MC sample has high ρ_{ext} value and small α_D and α_c values (causing small negative feedback), the QoI mean and standard deviation will have a larger absolute change, resulting in fluctuations in Figure 5.6.

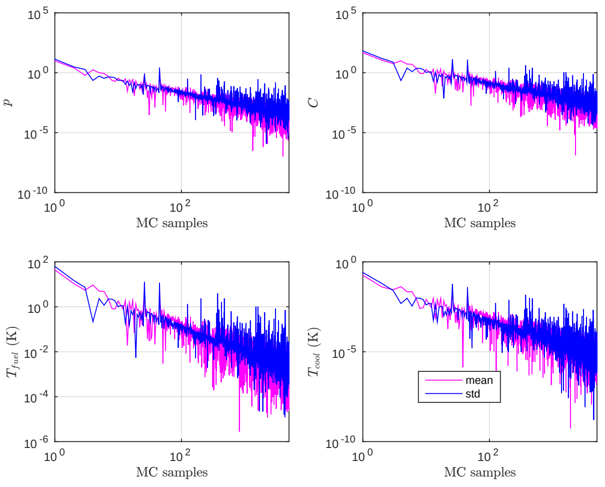


Fig. 5.6 Convergence of the mean values and standard deviations for four QoIs using MC sampling, Prior 1

Figure 5.7 shows the comparison of the mean values for four QoIs predicted by the MC sampling and PCE with order up to 5 for different priors. Here we only compare results at 21 time points which are uniformly distributed between 0 and 1 second. There is an excellent agreement of these results. PCE can accurately predict the mean values even at a very low expansion order.

Figure 5.8 shows the comparison of the standard deviations for four QoIs predicted by the MC sampling and PCE with order up to 5 for different priors. For p (normalized power) and C (delay neutron precursors) the PCE predicted standard deviations converge to the MC sampling values, while for T_{fuel} and T_{cool} PCE predictions converge to values that are slightly different than MC sampling standard deviations. They are likely to be caused by the PCE low order truncation in both inputs and outputs. The ODEs for T_{fuel} and T_{cool} include many material properties that depend on T_{fuel} and T_{cool} themselves, causing them to be more sensitive to small

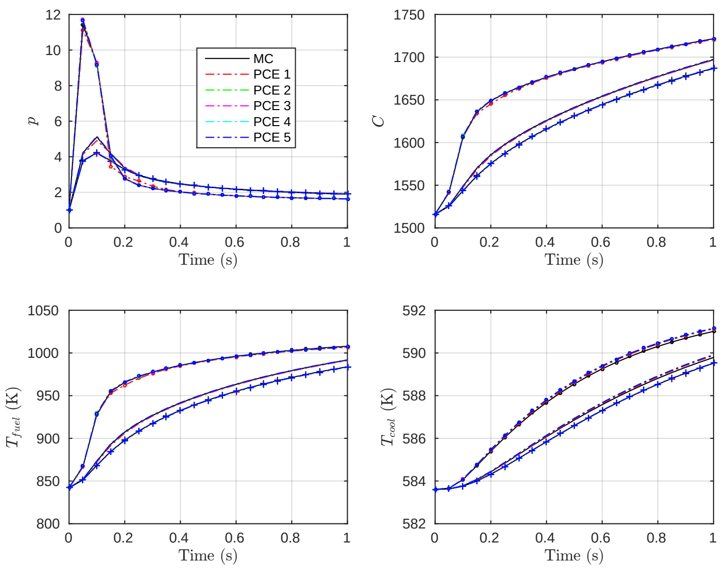


Fig. 5.7 Mean values of four QoIs by MC sampling and PCE (lines with no symbols are for Prior 1; lines with "+" symbols are for Prior 2; lines with "o" symbols are for Prior 3)

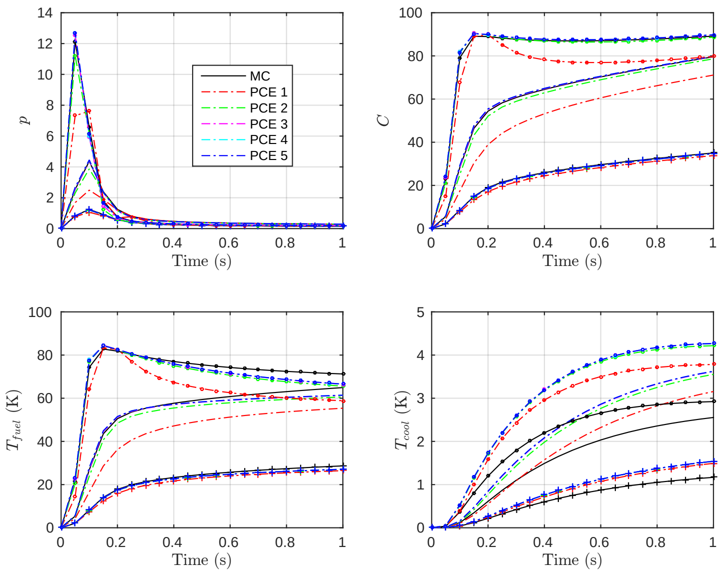


Fig. 5.8 Standard deviations of four QoIs by MC sampling and PCE (lines with no symbols are for Prior 1; lines with "+" symbols are for Prior 2; lines with "o" symbols are for Prior 3)

errors than p and C, which are only affected by T_{fuel} and T_{cool}. However, these discrepancies are very small compared to the mean values of those QoIs.

Note that PCE of order higher than 2 is already converged, their mean values and standard deviations are practically indistinguishable. Figure 5.9 shows the PCE modes of different PCE orders for the four QoIs at the peak power time ($\sim$0.1s) using Prior 1 (Prior 2 and Prior 3 have similar results). It is shown that:

1. Low-index of higher order PCE modes overlap with lower order PCE modes. For example, PCE of order 2 (pink symbols) have 10 terms, but the first 4 terms are the same as PCE of order 1 (blue symbols). Similar results can be observed for PCE of order 3-5.

2. The PCE modes are very close to 0 above order 2. This explains the fact that increasing the PCE order does not improve the results since all the higher-order coefficients are approaching 0.

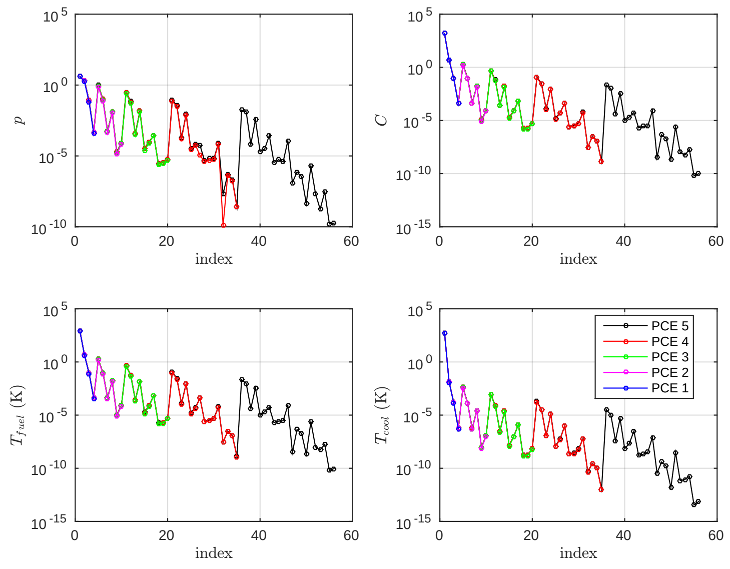


Fig. 5.9 PCE modes at peak power time for four QoIs, Prior 1

Comparing the mean values and standard deviations produced by PCE and MC sampling, it is found that even at a low order of 2, PCE produces mean values and standard deviations very close to the MC sampling. Here, the MC sampling requires solving the original system of ODEs 5000 times. The PCE technique, however, requires solving the new system of ODEs only once. The new system of ODEs, as mentioned before, is P times larger than the original system of ODEs, where $(P+1)$ can be found in Table 5.7. The simulation time for MC sampling with 5000 samples is 43 minutes, while the simulation time for PCE order 1 to 5 ranges from less than 1 second to 9 seconds.

Application of PCE for forward UQ means approximating model QoIs as polynomial functions of random input parameters. Such functions have demonstrated that they can successfully represent the stochastic nature of the outputs in terms of their mean values and standard deviations. The PCE essentially constitutes a metamodel of the original problem and can be used as surrogates during the inverse UQ process.

5.4 Inverse UQ Results

In this Section, we will demonstrate the application of PCE as metamodel for the inverse UQ process. First, synthetic experimental data of the fuel temperature T_{fuel} is introduced, which is intentionally designed to be different from simulation results. The purpose is to test if the inverse UQ process is capable of quantifying the uncertainties (mean, variances and PDFs) of the three random input parameters given the (synthetic) experimental data. Direct simulation is also performed to provide a reference solution.

5.4.1 Synthetic Experimental Data

Note that the σ term in the posterior definition is the standard deviation of the measurement error, and it can be different for each measurement points. The measurement error is assumed to be 0.5% of the measurement values. Since T_{fuel} mean values (Figure 5.7) are around $1000K$, this corresponds to $\pm 5K$ measurement noise, which is a realistic choice. Even though we expect this term to be reported with the experimental data, there can be a mistake in the reported value or in some cases it will be missing. Therefore, three different σ values are tested, they are $\sigma_{\text{exp},1} = 0.5\%$, $\sigma_{\text{exp},2} = 1.0\%$ and $\sigma_{\text{exp},3} = 0.25\%$.

Synthetic experimental data for T_{fuel} is shown in Figure 5.10, together with the simulation results at nominal values of the three random inputs from different priors. The synthetic experimental data is generated at $\rho_{\text{ext},0}$, $\alpha_{D,0}$ and $\alpha_{c,0}$ plus Gaussian random perturbation with standard deviation $\sigma_{\text{exp},1}$ (percentage of mean values).

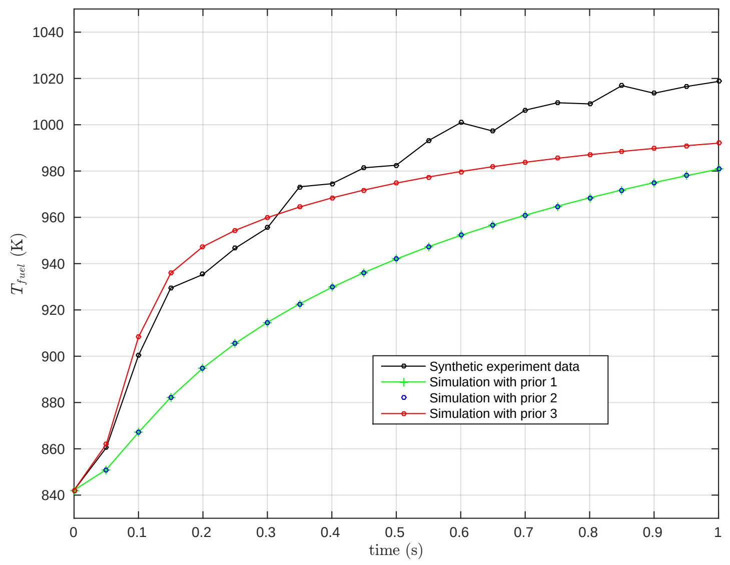


Fig. 5.10 Comparison of synthetic experimental data and simulation results with priors for T_{fuel} (prior 1 and prior 2 overlap)

5.4.2 Convergence of MCMC Samples

50,000 MCMC samples were generated by direct simulation, as well as PCE metamodels of order 1 to 5. Table 5.8 shows the time used to generate 50,000 MCMC samples. PCE surrogates only involve polynomial evaluations and the time is reduced by 2-3 orders of magnitude compared with the direct simulation. In practical situation, where a single direct simulation takes hours to days, the computational time will be reduced by much larger orders of magnitude.

Table 5.8 Time taken to generate 50,000 MCMC samples

Models used in MCMC	Time (minutes)
Direct simulation	425.0
PCE surrogate order 1	0.6
PCE surrogate order 2	1.2
PCE surrogate order 3	2.3
PCE surrogate order 4	4.3
PCE surrogate order 5	7.2

Table 5.9 shows all the 5 cases considered for inverse UQ and each case includes running PCE metamodels from order 1 to 5. By comparing cases A, B and C, we can evaluate the influence of using different σ_{exp}'s on the posterior. By comparing cases A, D and E, we can see the influence of using different priors on the posterior.

Table 5.9 Difference cases considered for MCMC sampling

Cases	Direct simulation	PCE surrogates
A	$\sigma_{\text{exp},1}$	Prior 1, $\sigma_{\text{exp},1}$
B	$\sigma_{\text{exp},2}$	Prior 1, $\sigma_{\text{exp},2}$
C	$\sigma_{\text{exp},3}$	Prior 1, $\sigma_{\text{exp},3}$
D	$\sigma_{\text{exp},1}$	Prior 2, $\sigma_{\text{exp},1}$
E	$\sigma_{\text{exp},1}$	Prior 3, $\sigma_{\text{exp},1}$

For each case (A - E) and each simulation type (full model direct simulation, or PCE metamodels of order 1 - 5), the first 10,000 of 50,000 MCMC samples are discarded for burnin. **Burnin** [44] is the practice of throwing away some iterations at the beginning of a Markov chain. Burnin is intended to give the Markov chain time to reach its equilibrium distribution, as starting from a "bad" point may over-sample regions that are actually very low probability under the equilibrium distribution. Because consecutive samples in a Markov chain are dependent, "thinning" is also performed to reduce auto-correlation among the samples,

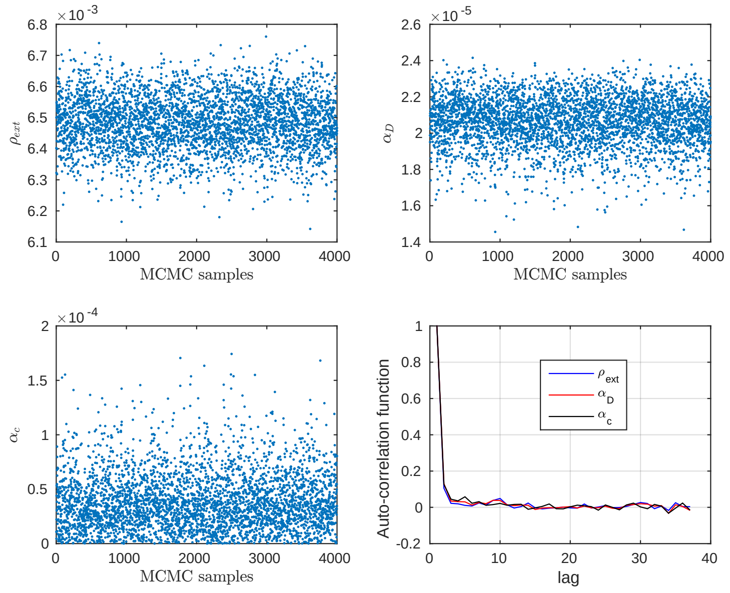


Fig. 5.11 Trace plots and auto-correlation functions of the Markov chain from direct simulation of Case A

which means discarding all but every k^{th} sample. For the current research, we choose $k = 10$ for thinning of the chain, leaving us with 4000 samples.

Figure 5.11 shows the trace plot and auto-correlation function (ACF) for the Markov chains for case A using direct simulation. The other Markov chains show similar behavior so they are not reported here. The trace plot shows a good mixing of the Markov chain. The fast decay of the Markov chain ACF after thinning also indicates a good mixing, which is consistent with the trace plot. Figure 5.12 shows the convergence of means and standard deviations of ρ_{ext}, α_D and α_c that is similar to Figure 5.6. Figure 5.12 shows that it is sufficient to use 50,000 MCMC samples because the mean values and standard deviations converged.

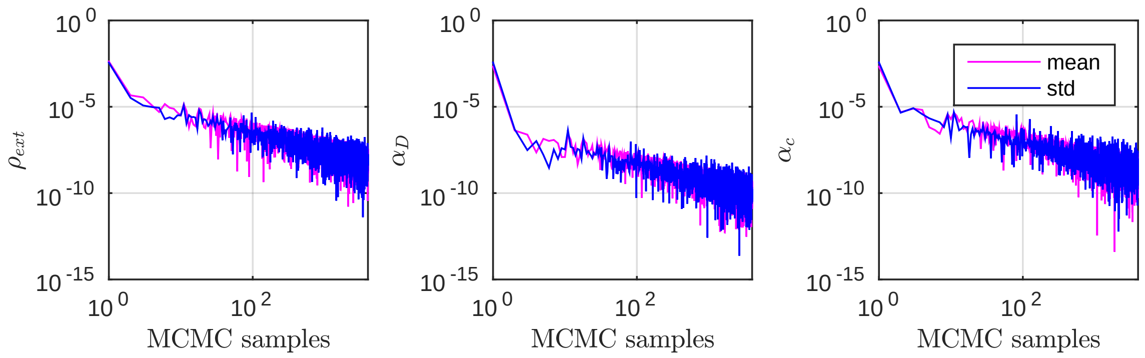


Fig. 5.12 Convergence of mean values and standard deviations of the Markov chain from direct simulation of Case A

5.4.3 Bayesian Solution for Posterior Statistical Moments

The MCMC samples (after burnin and thinning) will be analyzed to look at the statistical moments (mean values and standard deviations) and PDFs. Table 5.10 - 5.12 present the mean values and standard deviations for the posteriors of three random input parameters for all 5 cases. The same data sets are plotted in Figure 5.13 - 5.15. In all the tables and figures, "Direct" refers to full model direct simulation, and "PCE #" means using PCE metamodel of order #.

Table 5.10 Statistical moments of the posteriors for External reactivity insertion ρ_{ext}

Cases	moments	PC-1	PC-2	PC-3	PC-4	PC-5	Direct
Case A	mean	6.36E-03	6.32E-03	6.55E-03	6.48E-03	6.49E-03	6.48E-03
	std	1.58E-04	8.89E-05	9.83E-05	1.18E-04	9.60E-05	8.47E-05
Case B	mean	6.84E-03	6.18E-03	6.65E-03	6.44E-03	6.45E-03	6.38E-03
	std	2.98E-04	1.96E-04	2.87E-04	2.12E-04	2.00E-04	1.74E-04
Case C	mean	6.66E-03	6.32E-03	6.59E-03	6.54E-03	6.51E-03	6.52E-03
	std	1.21E-04	5.73E-05	5.15E-05	4.67E-05	4.37E-05	4.13E-05
Case D	mean	6.33E-03	6.52E-03	6.51E-03	6.53E-03	6.52E-03	6.46E-03
	std	2.46E-04	3.83E-05	4.38E-05	4.05E-05	4.42E-05	8.50E-05
Case E	mean	6.22E-03	6.42E-03	6.51E-03	6.52E-03	6.54E-03	6.50E-03
	std	1.10E-04	6.61E-05	6.12E-05	6.54E-05	6.52E-05	8.49E-05

Table 5.11 Statistical moments of the posteriors for Doppler reactivity coefficient α_D

Cases	moments	PC-1	PC-2	PC-3	PC-4	PC-5	Direct
Case A	mean	2.10E-05	2.01E-05	2.03E-05	1.97E-05	1.96E-05	2.06E-05
	std	3.40E-06	1.01E-06	1.06E-06	1.16E-06	1.97E-06	1.32E-06
Case B	mean	3.81E-05	1.76E-05	2.36E-05	1.88E-05	1.87E-05	1.82E-05
	std	5.28E-06	2.52E-06	5.54E-06	3.03E-06	2.30E-06	3.01E-06
Case C	mean	3.16E-05	1.99E-05	2.03E-05	2.08E-05	2.11E-05	2.16E-05
	std	4.05E-06	8.78E-07	5.35E-07	4.67E-07	5.15E-07	6.12E-07
Case D	mean	1.87E-05	2.18E-05	2.03E-05	2.05E-05	2.04E-05	2.02E-05
	std	1.24E-05	1.14E-06	1.08E-06	1.27E-06	1.21E-06	1.39E-06
Case E	mean	2.44E-05	2.18E-05	2.17E-05	2.20E-05	2.22E-05	2.08E-05
	std	1.51E-06	7.33E-07	7.98E-07	7.59E-07	7.68E-07	1.35E-06

The "true" values at which to generate the synthetic experimental T_{fuel} data are $\rho_{\text{ext},0} = 6.500 \times 10^{-3}$, $\alpha_{D,0} = 2.067 \times 10^{-5}$ and $\alpha_{c,0} = 5.342 \times 10^{-5}$. By analyzing the posterior mean values for ρ_{ext}, α_D and α_c, the following can be summarized:

1. The mean values of PCE metamodels generally approach the solutions of direct simulation at higher order.

Table 5.12 Statistical moments of the posteriors for coolant temperature coefficient α_c

Cases	moments	PC-1	PC-2	PC-3	PC-4	PC-5	Direct
Case A	mean	5.48E-04	3.00E-04	3.04E-05	4.17E-05	4.30E-05	3.87E-05
	std	8.38E-04	4.63E-04	1.19E-04	1.81E-05	3.17E-05	2.57E-05
Case B	mean	6.60E-02	1.64E-02	2.52E-03	5.44E-05	6.39E-05	9.67E-05
	std	8.04E-02	1.87E-02	4.56E-03	1.17E-04	3.01E-05	6.60E-05
Case C	mean	2.55E-03	1.96E-05	2.18E-05	2.35E-05	2.81E-05	1.46E-05
	std	2.87E-03	1.09E-05	1.59E-05	2.46E-05	7.96E-06	1.12E-05
Case D	mean	2.06E-02	5.98E-03	5.03E-04	3.31E-04	7.15E-05	4.59E-05
	std	4.34E-02	8.49E-03	8.75E-04	6.88E-04	1.15E-04	2.86E-05
Case E	mean	9.76E-04	7.25E-04	2.75E-04	1.58E-04	1.19E-04	3.68E-05
	std	1.10E-03	1.09E-03	3.24E-04	3.24E-04	1.66E-04	2.67E-05

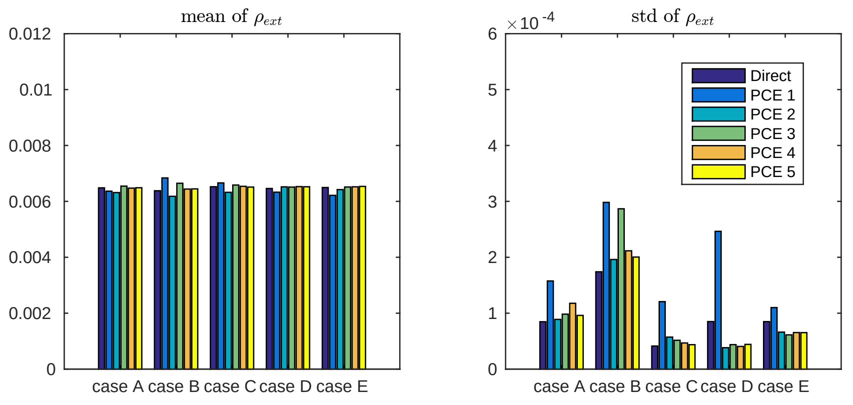


Fig. 5.13 Comparison of mean values and standard deviations for ρ_{ext}

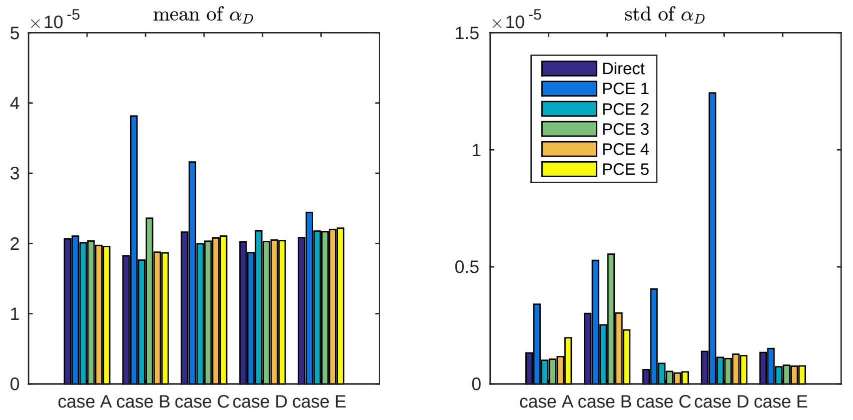


Fig. 5.14 Comparison of mean values and standard deviations for α_D

2. The mean values using different priors (cases A, D, E) are very close to each other, especially if we look at direct simulations and PCE surrogates of order higher than 3.

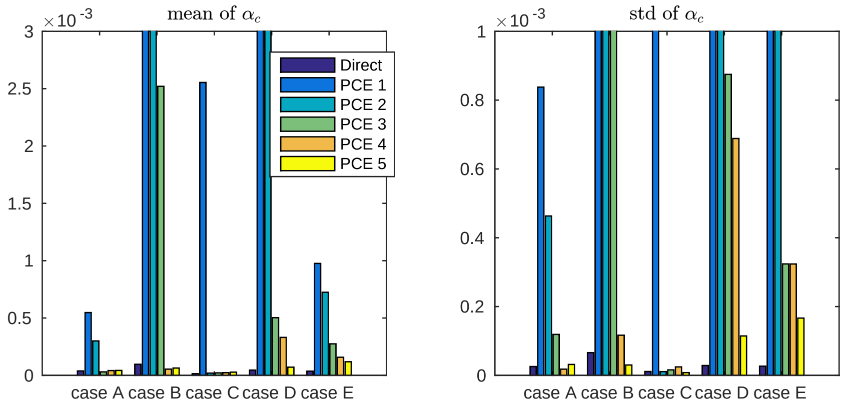


Fig. 5.15 Comparison of mean values and standard deviations for α_c

3. Solutions using different σ_{exp} values are also very close to each other, especially for ρ_{ext} and α_D. For α_c the posterior mean values using different σ_{exp} are less close.
4. Finally, for ρ_{ext} and α_D the mean values from inverse UQ are very close to "true" (synthetic) values of $\rho_{\text{ext},0}$ and $\alpha_{D,0}$. The results for α_c are not close to $\alpha_{c,0}$ for many cases. The reason is that the model is not sensitive to this parameter. The inverse UQ process is unable to quantify its value.

The agreement for the posterior standard deviations is less obvious. But again it can be observed that the results of direct simulation and high order PCE metamodels are close for ρ_{ext} and α_D, and less satisfactory for α_c.

5.4.4 Bayesian Solution for Posterior PDFs

Figures 5.16 - 5.20 show the posterior PDFs of ρ_{ext}, α_D and α_c, for cases A - E. This is another way to check if the posterior solutions with PCE surrogates will converge to the posterior solutions by direct simulation. If we can confirm this, real models that are computationally prohibitive (hours to days for a single simulation) can be confidently substituted by a metamodels constructed by the PCE or other stochastic spectral techniques.

Based on Figure 5.16 - 5.20, it is obvious that with the increasing of PCE order the solutions with PCE surrogates converge to the solution with direct simulation. There are some minor exceptions, and sometimes the convergence is not monotonic with PCE order. Nonetheless, the solutions by PCE surrogates up to order 5 demonstrate excellent accuracy and efficiency if we take the execution time into consideration.

An important question that remains is: what is the relationship between a prior and its corresponding posteriors? Figure 5.21 shows the comparison of priors 1 - 3 and posteriors A -

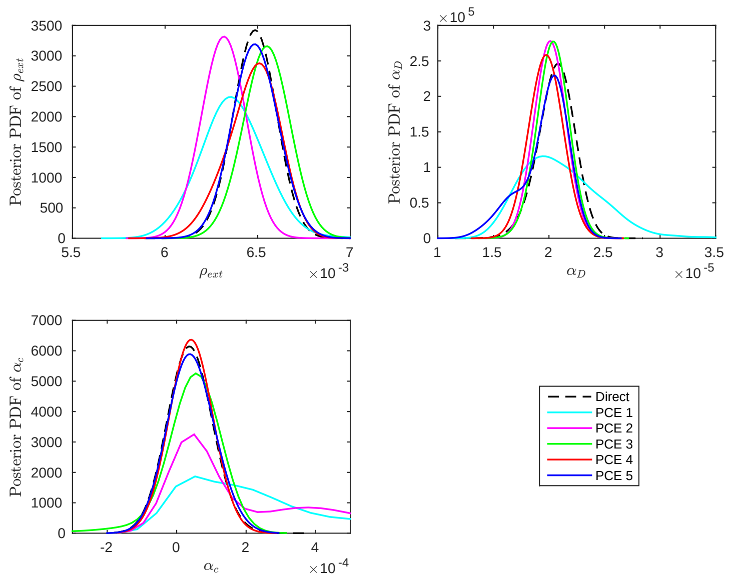


Fig. 5.16 Posterior PDFs for ρ_{ext}, α_D and α_c, case A

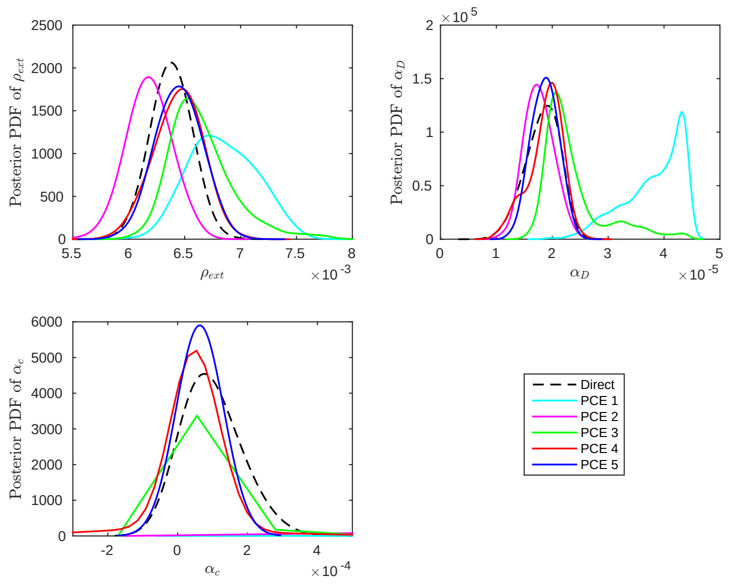


Fig. 5.17 Posterior PDFs for ρ_{ext}, α_D and α_c, case B

E. Posterior results are from PCE surrogate of order 5, which are close to results from direct simulations.

From Figure 5.21 it can be seen that:

- For all three parameters ρ_{ext}, α_D and α_c, posteriors A, D and E are very close. This is desirable because it means that we will arrive at the same posterior solution with the inverse UQ no matter what prior we start with. Therefore, whenever we want to do

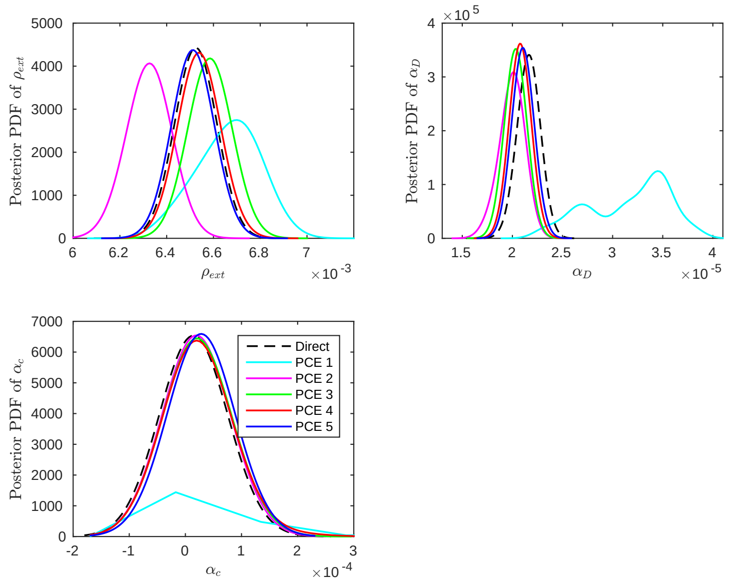


Fig. 5.18 Posterior PDFs for ρ_{ext}, α_D and α_c, case C

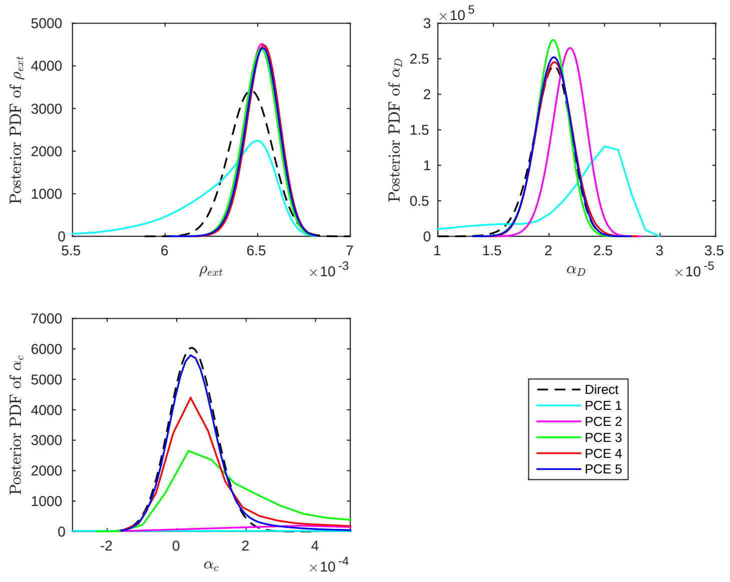


Fig. 5.19 Posterior PDFs for ρ_{ext}, α_D and α_c, case D

inverse UQ for a new model given experimental data, we can start with a prior that better reflect our ignorance about the input uncertainty.

- Posteriors A, B and C are different, indicating that σ_{exp} values do have important influence during the inverse UQ process. Larger σ_{exp} values ($\sigma_{\text{exp},2}$ for posterior B) results in larger posterior variance, while smaller σ_{exp} values ($\sigma_{\text{exp},3}$ for posterior C) results in smaller posterior variance.

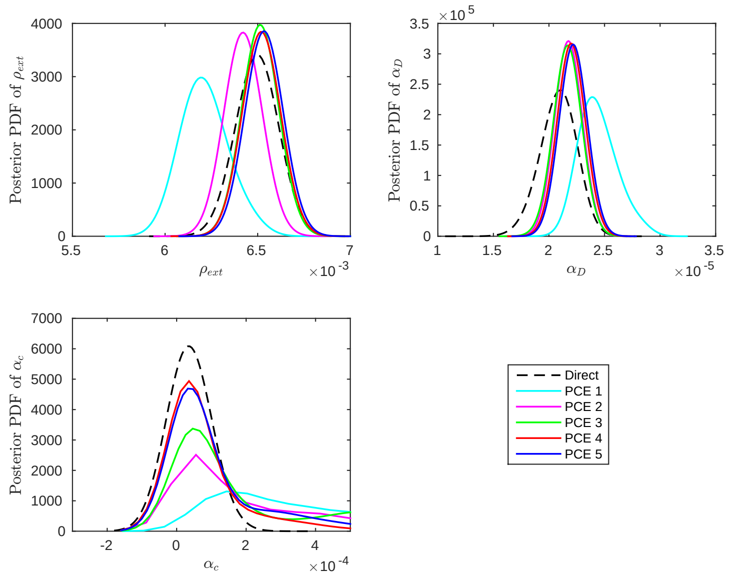


Fig. 5.20 Posterior PDFs for ρ_{ext}, α_D and α_c, case E

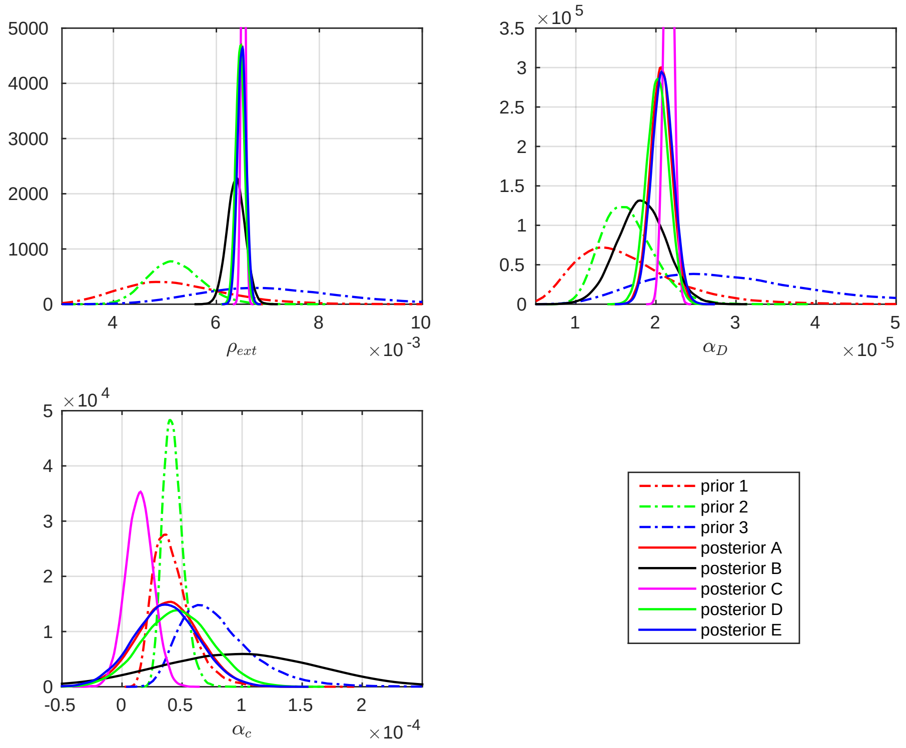


Fig. 5.21 Comparison of prior and posterior PDFs

- For ρ_{ext}, posterior uncertainties are much less than prior uncertainties (smaller variance), which means our ignorance about this parameter is greatly reduced. The results for α_D are less satisfactory (posterior B has large variance) but still acceptable.

- The results for α_c are not as good because the posteriors have larger variance than priors. It has been mentioned earlier in the chapter that the model is insensitive to this uncertain input parameter because the change in coolant temperature is very small. Inverse UQ process failed to reduce the uncertainty associated with α_c.

On the other hand, if the model is insensitive to certain uncertain input parameter, it is meaningless to try to reduce its uncertainty since the model can have close QoIs with a wide range of this parameter. Therefore, it is reasonable to hold a wider uncertainty range to better reflect our ignorance about this parameter.

5.5 Validation Results

Now that the posterior distributions for ρ_{ext}, α_D and α_c has been calculated by the inverse UQ process, they can be used instead of the priors (expert judgements) in a new forward UQ process. And this time we would like to see if the updated uncertainties (posterior A - E) in the three uncertain input parameters will produce outputs that agree better with the synthetic experimental data.

By looking at Figure 5.21, it is obvious that the posterior PDFs of all three random input parameters have shapes that are close to Gaussian distributions. Therefore, for the new forward UQ process (based on posterior), we can model the three input parameters as normal instead of lognormal. Again, we use PCE for the new forward UQ process since it has been demonstrated to have an excellent accuracy and efficiency. Note, that if we approximate normal random variable with Hermite polynomials, an expansion of order 1 will be exact:

$$I(\xi) = \sum_{i=0}^{P} I^i \Psi_i(\xi) = \sum_{i=0}^{1} I^i \Psi_i(\xi) = I^0 + I^1 \xi$$

The reason is that all we need to characterize a normal random variable is its mean value and standard variation, which are I^0 and I^1, respectively. Here ξ is a standard normal random variable $\xi \sim \mathcal{N}(0,1)$.

Figure 5.22 and Figure 5.23 show the mean values and standard deviations, respectively, for the new forward UQ process based on posteriors A - E. It is found that the mean values are very close for different posteriors. Standard deviations based on posterior A, D and E are close but they are very different with posterior B and C. This is because posterior B has larger variance for ρ_{ext}, α_D and α_c while posterior C has smaller variance.

Finally, Figure 5.24 shows the comparison of synthetic experimental data with the forward UQ results based on posteriors A - E. The dashed lines are the upper and lower error bounds

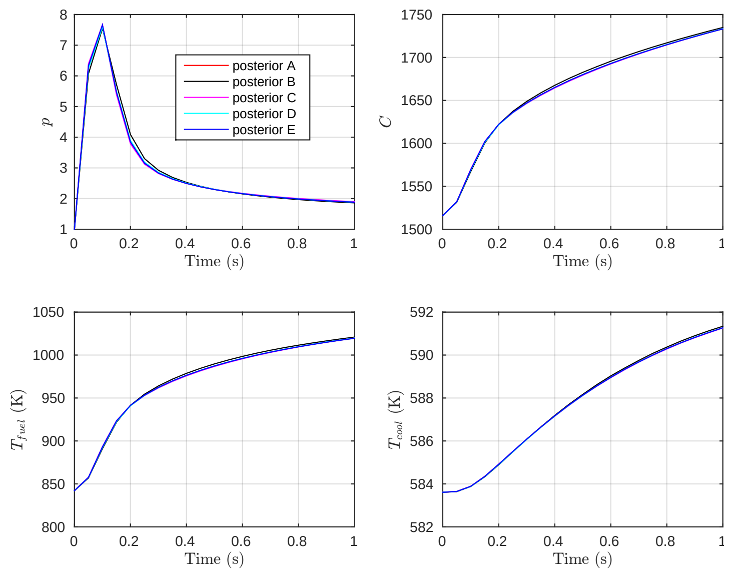


Fig. 5.22 Mean values of four QoIs based on posteriors from case A - E

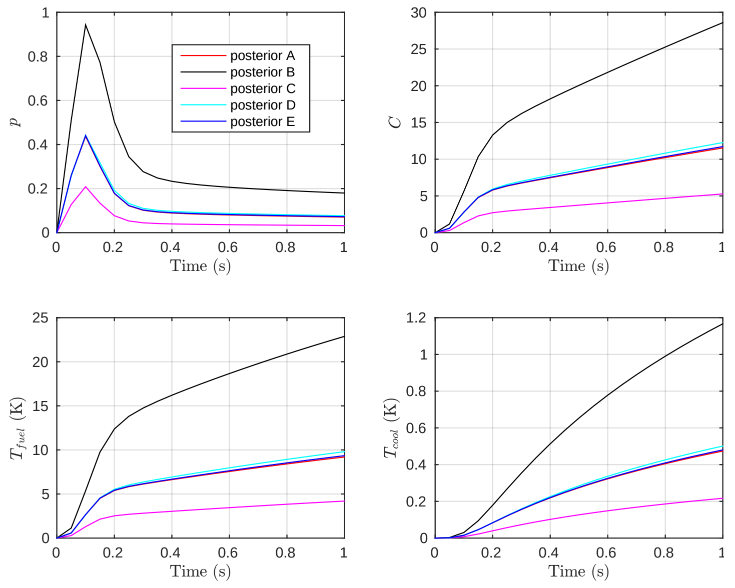


Fig. 5.23 Standard deviations of four QoIs based on posteriors from case A - E

which are calculated by adding and subtracting the standard deviations from the mean values. The following conclusion can be made about validation based on Figure 5.24:

1. By using the posterior uncertainties, the agreement between the mean values and the synthetic experimental data is greatly improved for all 5 cases.
2. The synthetic experimental data falls within one standard deviations of the simulation results for posteriors A, B, D and E. Posterior C result has the smallest standard deviation such that it fails to envelop some data points.

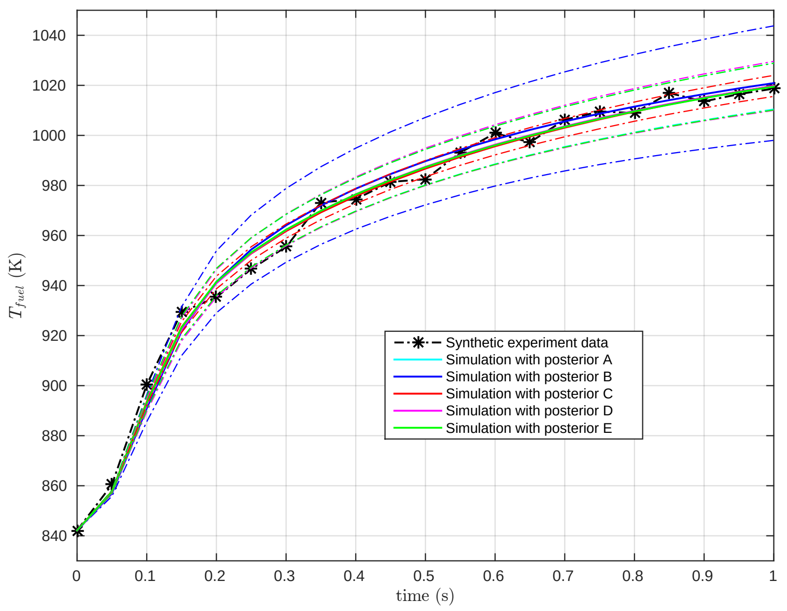


Fig. 5.24 Comparison of fuel temperature synthetic experimental data with simulation results based on posteriors

We can do another inverse UQ process, using the current posteriors as new priors until the uncertainty cannot be reduced any further. For a general uncertain PDE/ODE model, it is suggested to choose a less informative prior (one with larger variance) and repeat the inverse UQ process until the uncertainty in input parameters cannot be reduced.

5.6 Discussions

The application in this chapter has some limitations even though satisfactory results have been achieved:

1. Galerkin-PCE as surrogate model only works for special cases when we know the exact functional form the computer model. We have to re-code the model in an intrusive way. For complex real world applications when we need to treat the computer code as a black box, Galerkin-PCE cannot be used. However, NISP can be used to solve for the PCE coefficients, but we may not achieve a optimal rate of convergence.

2. The PRKE coupled with lumped parameter TH feedback model is only used to demonstrate the idea of inverse UQ in the Bayesian framework. For more sophisticated applications, the metamodeling technique will change, while the general procedure of performing inverse UQ remains the same.

Chapter 6

APPLICATION TO TRACE PHYSICAL MODEL PARAMETERS

In this chapter, the uncertainties associated with physical model parameters of system TH analysis code TRACE [138] will be quantified, based on steady-state void fraction data from the international OECD/NRC BWR Full-size Fine-Mesh Bundle Tests (BFBT) benchmark [98]. The inverse UQ process adopts metamodels built with Sparse Grid Stochastic Collocation (SGSC) method. Starting with 36 physical model parameters, we performed SA to identify 5 significant parameters for inverse UQ. This research addresses the problem of lacking uncertainty information about TRACE physical input parameters, which has been often ignored or described using expert opinion or personal judgment in prior uncertainty and SA work. The work presented in this chapter is also published in [152] [156] [158].

6.1 Problem Definition

For best-estimate system TH codes, significant uncertainties also come from the closure laws (also known as correlations or constitutive relationships) which are used to describe the transfer terms in the balance equations. For example, the physics portrayed in TRACE [138] is simulated by a two-fluid model for the two-phase flow, for which a number of constitutive and closure models must be used for model completeness. These physical models govern the mass, momentum and energy exchange between the fluid phases and surrounding medium (walls), varying according to the type of two-phase flow regime. When the closure models were originally developed, their accuracy and reliability were studied with a particular experiment. However, once they are implemented in a TH code as empirical correlations and used for prediction of different physical systems, the accuracy and uncertainty characteristics of these correlations are no longer known to the code user. Previously in the uncertainty and sensitivity study of such codes, physical model uncertainties are simply ignored, or described using expert opinion or self-judgment. This necessitates the work to accurately quantify the uncertainties in physical models of best-estimate system codes like TRACE [138] and RELAP5 [36].

6.1.1 Overview of System TH Analysis

In nuclear engineering, “system codes” are computer codes that are used to analyze complex reactor systems during normal operations as well as accident or transient conditions, especially for the TH analysis. A typical system TH code consists of the capabilities to model the reactor components, such as pipes, pressurizers, valves, and pumps, as well as multiple phase flows within the hydrodynamic models.

The history of system TH codes dates back to the beginning of the 1970s when first system code utilized the homogeneous equilibrium model with three balance equations to describe the two-phase flow [109]. Nowadays the most advanced system TH codes are built upon solutions of the so-called “two-phase two-fluid model” which results in at least six balance equations. The two-phase two-fluid model consists of mass, momentum and energy conservation equations for fluid and vapor phases separately supplemented by a suitable set of constitutive equations. The number of constitutive equations required to close the equation system is dependent on the number of balance equations. Furthermore, two-phase flows consist of different flow regimes based on their appearance and the flow structure. The regimes are used to select appropriate closure relationships to model heat transfer, interfacial drag, and other flow conditions.

Some of the most famous system TH codes are TRACE [138] (TRAC/RELAP Advanced Computational Engine), RELAP5 [36] (Reactor Excursion and Leak Analysis Program), ATHLET [16] (Analysis of THermal-hydraulics of LEaks and Transients), CATHARE [11] [33]. See reviews of system TH codes in [109] [116] [117]. Also, detailed comparison of the aforementioned system TH codes are presented with respect to their (1) conservation equations, flow regimes, numerics and significant assumptions [116], and (2) closure relations, validation, and limitations [117].

Major challenges for current system TH modeling are caused by our lack of understanding and proper techniques to model the interaction mechanism at the interface between the liquid and vapor phases. Empirical correlations are widely used to model the interfacial transfer mechanism (especially the interfacial momentum transfer). Consequently, substantial uncertainties can be propagate from these correlations to predictions of two-phase tow-fluid model.

6.1.2 TRACE Physical Model Parameters

TRACE [138] has been designed to perform best-estimate analyses of loss-of-coolant accidents (LOCAs), operational transients, and other accident scenarios in PWRs and boiling water reactors (BWRs). It can also model phenomena occurring in experimental facilities designed to simulate transients in reactor systems.

TRACE takes a component-based approach to modeling a reactor system. Each physical piece of equipment in a flow loop can be represented as some type of component, and each component can be further nodalized into a number of physical volumes (also called cells) over which the fluid, conduction, and neutron kinetics equations are averaged.

TRACE uses the two-fluid six-equation two-phase flow model that solves the conservation equations for mass, momentum and energy for the separate liquid and vapor phases of water with a common pressure field. For non-condensible phases in vapor, mixture equations for the conservation of momentum and energy are utilized. For these conservation equations, additional closure laws and constitutive relations are required to obtain a closed solution, which results in ten parameters that must be modeled: interfacial area, interfacial mass transfer, interfacial drag, liquid and vapor wall drag, interfacial liquid and vapor heat transfer, liquid-to-vapor sensible heat transfer coefficient, and the wall liquid and vapor heat transfer coefficients.

TRACE version 5.0 Patch 3 [138] includes options for user access to 36 physical model parameters from the input file. See Appendix C for a complete list of the 36 parameters and their descriptions. For forward uncertainty propagation, the users are free to perturb these parameters by addition or multiplication according to their personal or expert judgment. The work presented in this chapter will inversely quantify the uncertainties of the parameters relevant to the considered experimental data using SGSC metamodel by MCMC sampling. All quantified uncertainties will be multiplicative factors of the nominal values.

6.2 BFBT Benchmark

6.2.1 Overview of BFBT Benchmark

The international OECD/NRC BFBT [98] benchmark, based on the Nuclear Power Engineering Corporation (NUPEC) database, was created to encourage advancement in sub-channel analysis of two-phase flow in rod bundles, which has great relevance to the nuclear reactor safety evaluation. In the frame of the BFBT test program, single- and two-phase pressure losses, void fraction, and critical power tests were performed for steady-state and transient conditions. The BFBT benchmark has been widely used for uncertainty, sensitivity and validation studies.

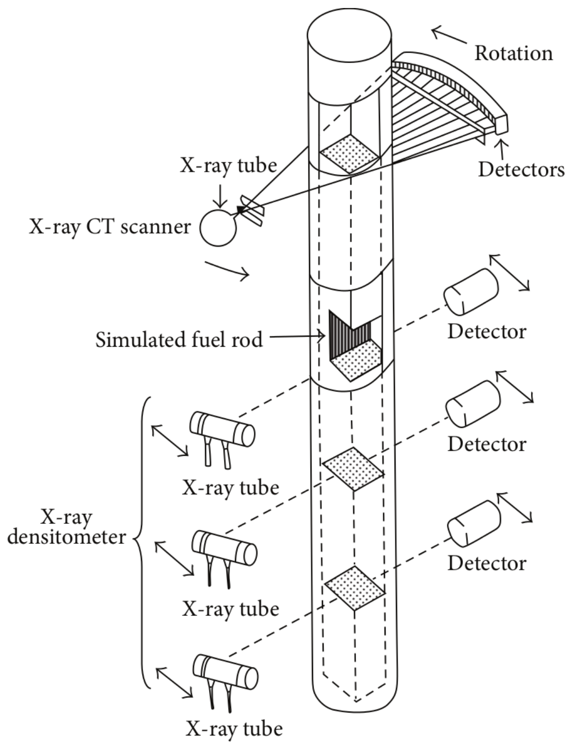


Fig. 6.1 Void fraction measurement structure.

The facility is full-scale BWR assembly, with measurement performed under typical reactor power and high-pressure, high-temperature fluid conditions found in BWRs. The full-scale fuel assembly inside the pressure vessel corresponds to the General Electric 8 × 8 assembly rod design, where each rod is electrically heated to simulate an actual reactor fuel rod. The heated length of the bundle corresponds to 3.7 m. Five different types of bundle assembly design with different combinations of geometries and power shapes were tested in the void distribution experiments.

Two types of void distribution measurement systems were employed: an X-ray computer tomography (CT) scanner and an X-ray densitometer. Under steady-state conditions, fine mesh void distributions were measured using the X-ray CT scanner located 50 mm above the heated length (i.e. at the assembly outlet). The X-ray densitometer measurements were performed at three different axial elevations from the bottom (i.e. 682 mm, 1706 mm and 2730 mm) under both steady-state and transient conditions. For the each of the four different axial locations, the cross-sectional averaged void fraction was also measured. Figure 6.1 shows the void fraction measurement facility and locations. The void fraction data will be used in the current study, and they will be referred to respectively from lower to upper positions as `VoidF1`, `VoidF2`, `VoidF3` and `VoidF4` in the following.

6.2.2 Selection of Experimental Data for Inverse UQ

The BFBT benchmark contains 392 steady-state void distribution test cases. For the current study, it is not practical to use all the test cases (each test case consists of 4 measurements, 1 at each of 4 axial elevations). Starting from the 86 test cases in assembly 4, we select the test cases by the following criteria:

1. Remove all the tests with negative void fraction data;
2. Remove all the tests that have lower void fractions at higher elevations;
3. Only keep one set of any duplicated tests;
4. Only keep measurements performed at high pressure and high power (above 7 MPa and 3MW), as these combinations will produce high void fractions thus more accurate metamodel.
5. Remove all the tests that have low void fraction (less than 1%) at low elevations;

Only 8 test cases satisfy these criteria and are selected for inverse UQ, their process conditions and void fraction data are included in Table 6.1.

Table 6.1 Process conditions and void fraction data for 8 selected cases of assembly 4

Test ID	Pressure (MPa)	Flow rate (t/h)	Inlet subcooling (kJ/kg)	Power (MW)	`VoidF1` (%)	`VoidF2` (%)	`VoidF3` (%)	`VoidF4` (%)
4101-58	7.152	54.58	50.6	3.52	5.80	43.4	63.4	64.5
4101-59	7.190	54.57	52.1	4.88	17.4	56.7	73.5	73.7
4101-60	7.178	54.62	50.5	4.89	17.3	56.8	73.3	74.0
4101-61	7.180	54.65	52.5	6.48	29.0	66.7	79.8	80.7
4101-67	7.248	69.58	54.6	4.48	4.50	42.1	63.0	66.8
4101-68	7.275	69.56	56.0	6.22	14.9	56.5	72.7	75.1
4101-84	8.680	54.66	53.2	3.35	3.80	37.4	57.9	60.2
4101-86	8.705	54.59	54.2	4.62	13.5	52.8	69.7	69.8

6.2.3 Void Fraction Data Correction

The X-ray densitometers can only capture the void fraction between the rod rows, therefore the measured data only shows the void fraction of a limited area of the subchannel. However, void fraction in the subchannel is not equally distributed as pointed out in [53]. For example, at low void fraction with bubbly flow, the void is concentrated in small bubbles close to the heat surface, while at high void fractions with slug flow, large bubbles are more likely to be located in the subchannel center. Consequently, the void fractions are under predicted at low void fractions and over predicted at high void fractions with the present X-ray densitometers. To resolve this issue, data correction has been suggested [53] and applied [67]. The correction formulas are proposed in [53], and the correction for assembly 4 data is shown in Equation 6.1.

$$\alpha_{\text{corrected}} = \frac{\alpha_{\text{measured}}}{1.167 - 0.001 \cdot \alpha_{\text{measured}}} \tag{6.1}$$

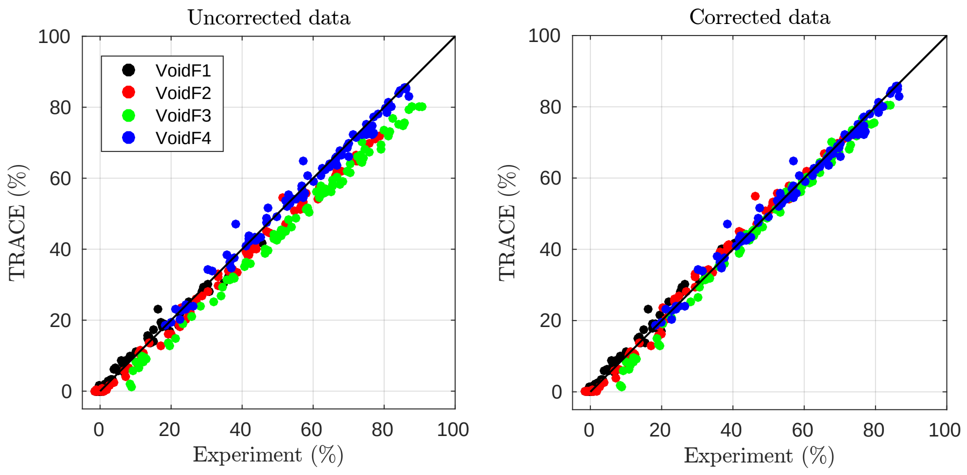


Fig. 6.2 Comparison of void fractions from BFBT measurement and TRACE simulation for all 86 test cases of assembly 4

All the void fractions are in (%) and Equation 6.1 is recommended for measured void fractions between 20% and 90% (note that `VoidF4` is not corrected because it is measured by CT scanner which does not have the aforementioned limitations of an X-ray densitometer).

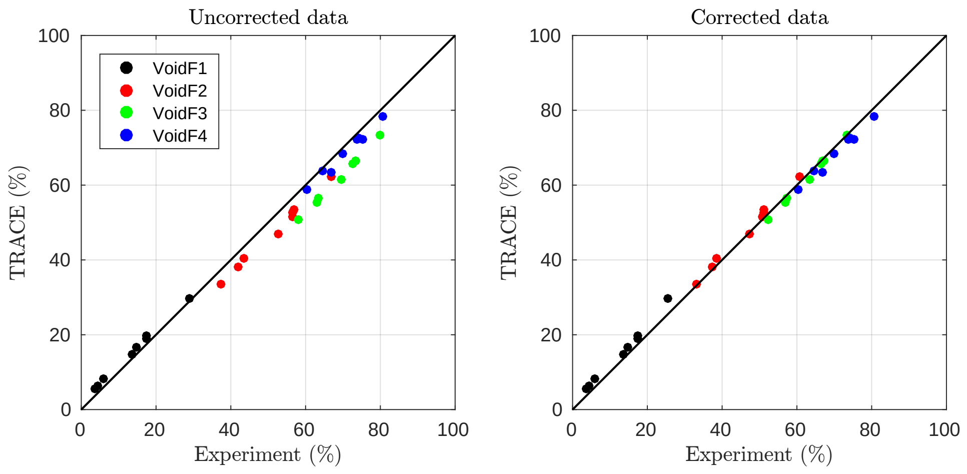


Fig. 6.3 Comparison of void fractions from BFBT measurement and TRACE simulation for 8 selected test cases of assembly 4

Figure 6.2 shows a comparison of void fraction from BFBT measurements (with and without correction) and TRACE simulations. All 86 test cases are presented. Before data correction, the majority of the void fractions are under predicted especially for `VoidF1`, `VoidF2` and `VoidF3`. After data correction, the data points are more concentrated close to the diagonal line, meaning that the agreement between measurement (BFBT) and calculation (TRACE) is improved. Figure 6.3 shows the comparison of the 8 selected cases. The improvement in the data is clear with void fraction correction for the selected cases.

6.2.4 Outline of Workflow

The workflow of follow-up study is outlined below:

1. Starting with 36 physical model parameters, perform a centered parameter study to remove parameters that are not utilized by TRACE.
2. Perform global SA for new (relevant) list of parameters, using Sobol' indices and correlation coefficients. Some parameters are expected to have much lower effect compared to others; these parameters will be removed.
3. The above two steps calculate the final list of physical model parameters whose uncertainty will be studied in the inverse UQ process.
4. Build SGSC metamodel of TRACE with selected input parameters.
5. Validate the metamodels.
6. Replace TRACE with the metamodel in MCMC sampling for the inverse UQ process.
7. Perform convergence diagnostics of MCMC chains and study the posterior distributions of selected physical model parameters.

6.3 Results for Global SA

As none of the methods for calculating Sobol' indices can effectively treat problems with dimensions as high as 36, we need to remove some parameters through a preliminary reduction study [1]. Many of the closure models are not relevant to the BFBT benchmark and will not be called by TRACE. For example, stratified flow (parameter `P1003` and `P1007`) and reflooding (parameter `P1034` and `P1035`) do not occur in the BFBT benchmark experiment.

6.3.1 Centered Parameter Study

This preliminary selection was done by centered parameter study, in which each parameter was perturbed a few steps around the nominal value (which is 1.0) one-by-one. DAKOTA [1] was used here to perturb each parameter 10 steps above and below nominal values with a step size of 0.02. The void fraction variance was calculated for each parameter. As expected, most of the variances are 0 or very close to 0. Ultimately, only 8 parameters produce variances larger than 10^{-3} for at least one of the output parameters `VoidF1`, `VoidF2`, `VoidF3` and `VoidF4`. These 8 parameters are shown in Table 6.2.

Table 6.2 List of 8 selected physical model parameters selected after centered parameter study

Parameter	Description
`P1008`	Single phase liquid to wall HTC
`P1009`	Single phase vapor to wall HTC
`P1012`	Subcooled boiling HTC
`P1013`	Nucleate boiling HTC
`P1022`	Wall drag coefficient
`P1023`	Form loss coefficient
`P1028`	Interfacial drag (bubbly/slug Rod Bundle - Bestion) coefficient
`P1029`	Interfacial drag (bubbly/slug Vessel) coefficient

6.3.2 Sobol' Indices and Correlation Coefficients

For the 8 selected physical model parameters, global SA was performed with both Sobol' indices and PCC/SRCC [158]. The Sobol' indices are calculated using PCE method in DAKOTA [1]. Table 6.3 and 6.4 show the Sobol' indices (both main and total effects) and PCC/SRCC of 8 selected parameters, for void fractions at four different elevations. Figure 6.4 and 6.5 visualize the data in Table 6.3 and 6.4, respectively.

Several major conclusions can be drawn from the tables and figures:

1. `P1009`, `P1013` and `P1023` have negligible Sobol' indices and their PCC and SRCC are also very small, indicating that void fraction is not sensitive to those parameters. They will be removed from further study.
2. Positive correlation coefficients indicate that void fraction increases with this parameter, and vice versa.
3. `P1029` is only important for `VoidF4`.

Table 6.3 Sobol' indices (main and total effects) for 8 selected physical model parameters

Parameter	`VoidF1`		`VoidF2`		`VoidF3`		`VoidF4`	
	main	total	main	total	main	total	main	total
`P1008`	8.51E-02	8.56E-02	9.10E-02	9.42E-02	1.20E-02	1.20E-02	1.17E-03	1.17E-03
`P1009`	4.02E-28	4.17E-28	5.93E-25	6.53E-25	5.50E-25	5.61E-25	4.66E-25	4.84E-25
`P1012`	9.12E-01	9.13E-01	1.28E-01	1.31E-01	1.20E-02	1.20E-02	1.16E-03	1.16E-03
`P1013`	4.05E-28	4.20E-28	5.20E-25	5.81E-25	5.46E-25	5.58E-25	4.64E-25	4.84E-25
`P1022`	1.13E-03	1.14E-03	2.07E-01	2.07E-01	2.79E-01	2.79E-01	6.93E-01	6.94E-01
`P1023`	4.22E-11	4.26E-11	2.13E-09	2.45E-09	1.38E-09	1.98E-09	1.80E-07	1.82E-07
`P1028`	9.52E-04	9.86E-04	5.71E-01	5.71E-01	6.97E-01	6.97E-01	7.79E-02	7.81E-02
`P1029`	1.58E-07	1.59E-07	6.90E-06	6.94E-06	1.72E-06	1.73E-06	2.25E-01	2.26E-01
Sum	0.9994	1.0006	0.9967	1.0033	0.9997	1.0003	0.9990	1.0010

Table 6.4 PCC and SRCC for 8 selected physical model parameters

Parameter	PCC				SRCC			
	`VoidF1`	`VoidF2`	`VoidF3`	`VoidF4`	`VoidF1`	`VoidF2`	`VoidF3`	`VoidF4`
`P1008`	-0.2897	-0.2907	-0.0946	-0.0291	-0.2801	-0.2873	-0.0970	-0.0247
`P1009`	0.0047	0.0084	0.0041	0.0113	0.0053	0.0186	0.0127	0.0123
`P1012`	-0.9497	-0.3511	-0.1055	-0.0259	-0.9555	-0.3290	-0.0917	-0.0298
`P1013`	0.0058	-0.0037	-0.0048	-0.0014	0.0059	-0.0014	-0.0069	-0.0067
`P1022`	-0.0160	-0.4499	-0.5271	-0.8320	-0.0207	-0.4395	-0.5076	-0.8395
`P1023`	-0.0013	0.0064	0.0052	-0.0022	-0.0009	0.0089	0.0105	0.0007
`P1028`	0.0267	0.7548	0.8334	0.2772	0.0306	0.7568	0.8377	0.2581
`P1029`	0.0045	-0.0058	-0.0039	0.4743	0.0084	-0.0035	-0.0062	0.4534

4. The sensitivity ranking for each of the parameters is mostly consistent between Sobol' indices and PCC/SRCC.
5. Sobol' indices are better measures of sensitivity than correlation coefficients, as they directly represent the part of output variance that can be attributed to each parameter. PCC/SRCC cannot consistently reflect this relative importance as well as Sobol' indices. For example, Sobol' indices show that only `P1022` and `P1028` are important for `VoidF3`, while PCC/SRCC show that `P1008` and `P1012` should be included, too.

Taking into account the details of each parameter, the observed sensitivity ranking can be explained as below:

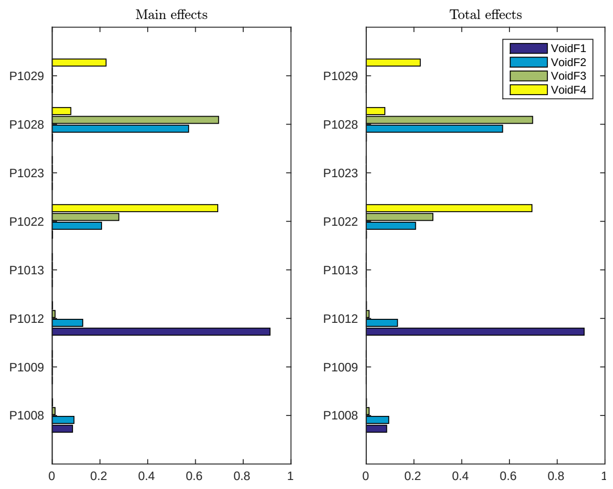


Fig. 6.4 Sobol' indices (main and total effects) for 8 physical model parameters.

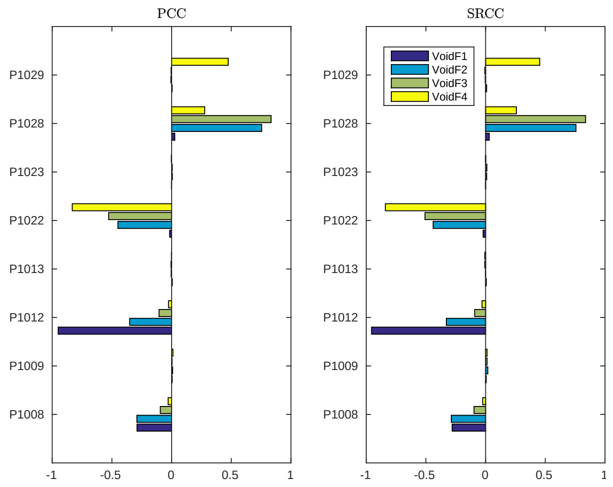


Fig. 6.5 PCC and SRCC ranking for 8 physical model parameters.

1. The significance of `P1008` (single phase liquid to wall HTC) decreases to almost zero at higher elevations. This is because single-phase liquid exists only in the lower elevations of the bundle.
2. Similarly, `P1012` (subcooled boiling HTC) is only important at lower elevations because this is where subcooled boiling occurs.
3. `P1022` (wall drag coefficient) increases at higher elevations.
4. `P1028` (interfacial drag bundle coefficient) dominates at intermediate locations.

Finally, the sum of main effects for all four QoIs is very close to 1.0, meaning that the interacting effects are negligible for the current study. Simply investigating the main effects and total effects are sufficient.

6.4 Sparse Grid Stochastic Collocation Metamodels

6.4.1 Constructing the Metamodels

In this work, we use the SG module of the Toolkit for Adaptive Stochastic Modeling And Non-Intrusive Approximation (TASMANIAN), developed at Oak Ridge National Laboratory [131] [132]. To construct the SGSC surrogate models we need the range of each physical model parameter, which is defined by the prior distributions and will be discussed later. Clenshaw-Curtis rule and a more recently developed $\mathcal{R}$-leja rule [21] are used as the one-dimensional building block.

The growth of the nodes in $\mathcal{R}$-leja rule is linear rather than exponential as in Clenshaw-Curtis rule. For five-dimensional surrogate model, Leja rule of precision 4 requires 126 nodes, while Clenshaw-Curtis rule of precision 3 and 4 require 145 and 301 nodes, respectively. Here, "precision" means that the underlying one-dimensional rule can exactly interpolates polynomials of a degree up to and including the precision value. Figure 6.6 illustrates the adopted sparse grids in two dimensions.

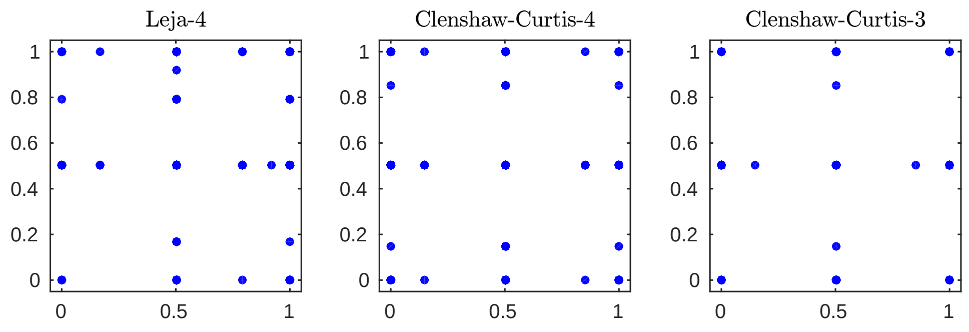


Fig. 6.6 Demonstration of two-dimensional nodes for Leja rule (precision 4), Clenshaw-Curtis rule (precision 4) and Clenshaw-Curtis rule (precision 3)

6.4.2 Validating the Metamodels

The metamodels constructed by SGSC method has to be validated before it can be applied in the inverse UQ process. It should be able to reproduce TRACE simulation results over a wide range of the 5 physical model parameters. DAKOTA is used to generate 100 samples using LHS of the input parameters for each of the 8 test cases and then act as a driver to use TRACE to calculate those samples. The metamodel was also evaluated with the DAKOTA-generated samples of input parameters to obtain the void fraction prediction. Table 6.5 presents the maximum absolute error in void fraction prediction among 100 samples for each test case. All of the metamodels are able to reproduce the TRACE void fraction results accurately. Leja-4 sparse grid rule is used for the inverse UQ process, as it utilizes the least number of TRACE runs for the construction of the metamodel.

Table 6.5 Validation results for metamodel constructed from different sparse grid rules

Metamodel	Test ID	Maximum absolute error			
		`VoidF1`	`VoidF2`	`VoidF3`	`VoidF4`
Leja-4	4101-58	1.02E-03	1.08E-03	3.18E-05	5.44E-04
	4101-59	2.32E-03	3.25E-04	8.64E-05	1.66E-04
	4101-60	1.64E-03	2.75E-04	8.50E-05	1.64E-04
	4101-61	4.57E-03	2.35E-04	3.05E-05	1.57E-04
	4101-67	9.42E-04	1.51E-03	4.80E-05	5.11E-04
	4101-68	6.38E-04	5.06E-04	1.53E-04	1.97E-04
	4101-84	7.51E-04	1.65E-03	2.89E-05	6.33E-04
	4101-86	5.22E-04	7.13E-04	2.79E-05	2.44E-04
Clenshaw-Curtis-4	4101-58	8.01E-04	3.05E-04	1.99E-05	3.28E-04
	4101-59	6.48E-04	1.12E-04	4.29E-05	3.32E-05
	4101-60	1.44E-03	9.35E-05	4.02E-05	3.36E-05
	4101-61	1.49E-03	7.42E-05	1.20E-05	6.67E-05
	4101-67	9.88E-04	3.27E-04	4.01E-05	2.90E-04
	4101-68	2.48E-04	1.81E-04	3.08E-05	3.57E-05
	4101-84	7.18E-04	2.96E-04	1.92E-05	5.49E-04
	4101-86	2.33E-04	1.92E-04	2.94E-05	4.96E-05
Clenshaw-Curtis-3	4101-58	7.68E-04	1.12E-03	6.16E-05	5.44E-04
	4101-59	1.78E-03	3.40E-04	6.58E-05	2.77E-04
	4101-60	1.80E-03	2.83E-04	6.69E-05	2.75E-04
	4101-61	3.20E-03	1.90E-04	5.01E-05	2.45E-04
	4101-67	8.67E-04	1.57E-03	6.88E-05	4.95E-04
	4101-68	6.40E-04	5.85E-04	9.59E-05	3.02E-04
	4101-84	6.22E-04	1.74E-03	5.49E-05	5.79E-04
	4101-86	5.52E-04	7.49E-04	5.20E-05	3.32E-04

6.5 Results for Inverse UQ

The measurement error is assumed to be 5% of BFBT void fraction data, which is needed for the variance term in the posterior formulation.

6.5.1 Selection of the Prior Distributions

Non-informative uniform priors are used in this study to reflect our ignorance with respect to the input parameters. These uniform priors will be a constant in the posterior function (Equation 4), and their ranges are important in building the SGSC metamodel. Intuitively priors with wide ranges should be used, but these will require much more nodes to build the metamodel. We decide the prior ranges using an iterative process: (1) begin with a wide range ($[0, 10]$) for each parameter, and build a low-precision metamodel; (2) adjust the prior ranges according to the posterior samples. If the samples for a certain parameter concentrate on a smaller range, narrow down its prior range accordingly. Otherwise if many posterior samples cluster on the boundary of its prior, increase the prior range; (3) build higher-precision metamodel on adjusted prior ranges. Table 6.6 shows the final chosen prior ranges for different physical model parameters.

Table 6.6 Prior uniform distribution ranges

Parameter	Uniform ranges
P1008	[0.5, 2.5]
P1012	[0.5, 1.5]
P1022	[0.0, 5.0]
P1028	[0.0, 4.0]
P1029	[0.0, 10.0]

6.5.2 MCMC Convergence Diagnostics

Algorithms 1 - 3 in Section 2.4 are used for MCMC sampling. 100,000 samples are generated for each case. All three algorithms take about 72 core-minutes to produce 100,000 samples using a current generation Intel CPU, which would otherwise take about 1167 core-hours (48 core-days) using direct TRACE simulation with the same processor. The first 10,000 samples are discarded as burn-in and then every 20th sample was kept from the remainder for thinning of the chain, leaving us with 4500 samples. Thinning is performed to reduce auto-correlation among the samples, which is shown in Figure 6.7.

Figure 6.7 shows mixing for the five parameters and the decay of the auto-correlation function of the Markov chain for Algorithm 3 (Markov chains for Algorithm 1 and 2 show very similar behavior). Note that all the five parameters are enforced to be positive, as negative values would not be physical. Very good mixing can be identified, and the auto-correlations for all the five parameters decay quickly after thinning, lending confidence that the Markov chain has converged. Figure 6.8 shows the convergence of mean values and standard deviations for each parameter. All mean values and standard deviations approach a constant value, also indicating the convergence of the Markov chain.

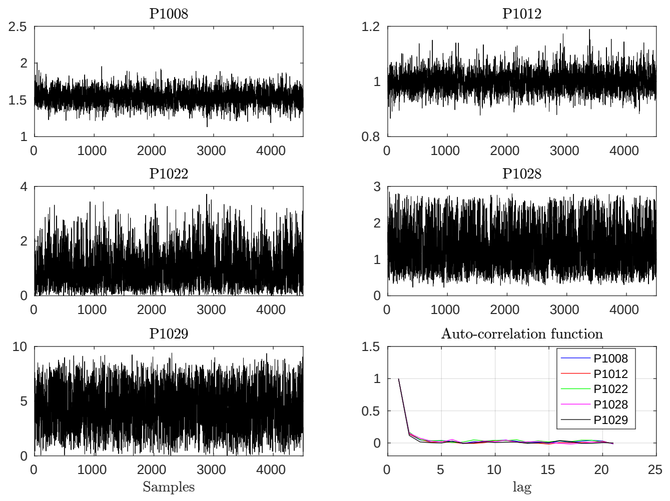


Fig. 6.7 MCMC chain trace plots and auto-correlation functions

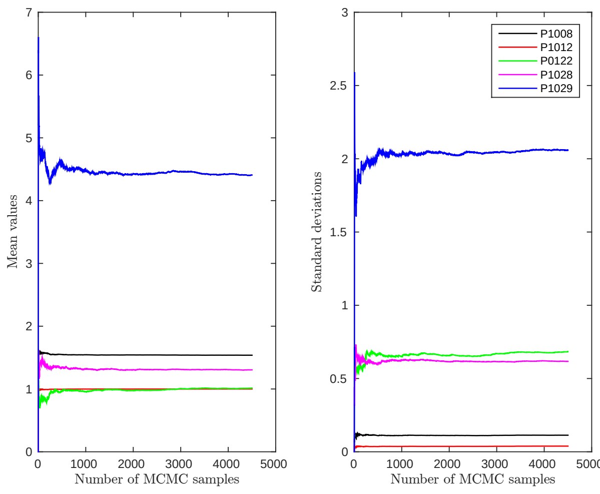


Fig. 6.8 Convergence of mean values and standard deviations

6.5.3 Investigating the MCMC Samples

Table 6.7 presents the statistics of MCMC chains for each parameter. The mean values and standard deviations from different MCMC algorithms are very similar for each parameter. In the following analysis the Markov chain from Algorithm 3 will be used, as it is representative of the other two Markov chains.

Table 6.7 MCMC chain statistics for different physical model parameters. Chain (1) - (3) are from adaptive MCMC Algorithms 1 - 3, respectively.

Parameter	Chain 1		Chain 2		Chain 3	
	mean	std	mean	std	mean	std
`P1008`	1.5359	0.1115	1.5315	0.1159	1.5384	0.1141
`P1012`	1.0005	0.0387	1.0015	0.0399	1.0000	0.0387
`P1022`	1.0113	0.6634	1.0478	0.6937	1.0095	0.6842
`P1028`	1.3000	0.6084	1.2861	0.6126	1.3046	0.6128
`P1029`	4.4053	2.0451	4.3410	2.0374	4.4119	2.0570

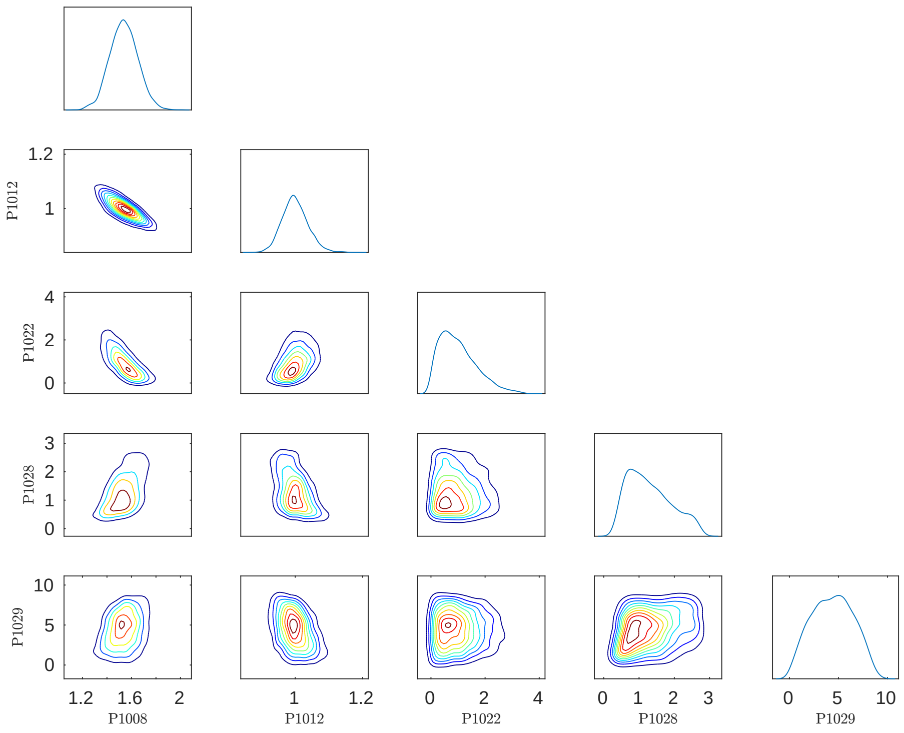


Fig. 6.9 MCMC chain pairwise joint density contours and marginal densities

Figure 6.9 shows the plot for pairwise joint density contours and marginal densities for the five physical model parameters. The marginal PDFs are evaluated using Kernel Density Estimation (KDE). This plot is useful for identifying potential correlation between the parameters. Highly linear correlations are observed between some parameters, such as `P1008` (single phase liquid to wall HTC) and `P1012` (subcooled boiling HTC). This indicates that in future forward uncertainty propagation studies, these input parameters should be sampled jointly, not independently, so that their correlation is captured. Table 6.8 shows the correlation coefficient matrix of all the parameters.

Table 6.8 Correlation matrix

Parameter	P1008	P1012	P1022	P1028	P1029
P1008	1.00				
P1012	-0.84	1.00			
P1022	-0.73	0.39	1.00		
P1028	0.52	-0.45	-0.06	1.00	
P1029	0.29	-0.39	-0.02	0.27	1.00

6.5.4 Fitted Posterior Distributions

To make these results more applicable to our eventual forward uncertainty propagation, we need to fit posterior samples to well-known distributions, such as Normal or log-normal distributions, so that they will be more easily sampled. From the marginal PDFs in Figure 6.9, it is obvious that for P1008, P1012 and P1029, we can consider these parameters as normal random variables. For P1022 and P1028, the PDFs are more skewed toward 0. Natural choices for these distributions include Gamma and log-normal. Gamma distributions are chosen because log-normal distribution has a longer tail which results in a poor match with these samples.

Table 6.9 Fitted distribution for each physical model parameter

Parameter	Distribution type	Distribution parameter 1	Distribution parameter 2
P1008	Normal	$\mu = 1.5377$	$\sigma = 0.1131$
P1012	Normal	$\mu = 1.0001$	$\sigma = 0.0386$
P1022	Gamma	$\alpha = 1.7581$	$\beta = 0.5767$
P1028	Gamma	$\alpha = 4.3106$	$\beta = 0.3027$
P1029	Normal	$\mu = 4.4080$	$\sigma = 2.0594$

Figure 6.10 and Table 6.9 show the fitted distribution for each physical model parameter and the parameters associated with each distribution, i.e. mean (μ) and standard deviation (σ) for normal distribution, shape α and scale β parameter for Gamma distribution. All the fitted distributions are accepted by Kolmogorov–Smirnov test at the 5% significance level. Figure 6.11 shows that good agreement can be achieved between the empirical cumulative distribution function (CDF) and fitted CDF for every parameter.

The fitted normal distribution for P1029 has very large standard deviation and a mean value that deviates significantly from the nominal value. This is most likely because the TRACE model for BFBT benchmark is insensitive to P1029 (as shown in Section 6.3, P1029 only affects VoidF4), therefore our ignorance with regards to P2019 cannot be reduced by using

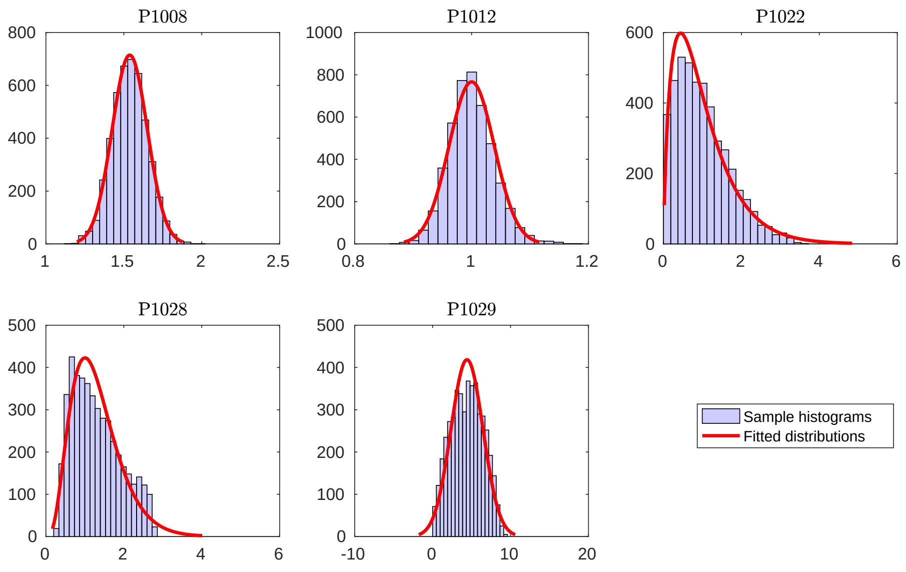


Fig. 6.10 Fitted posterior probability densities

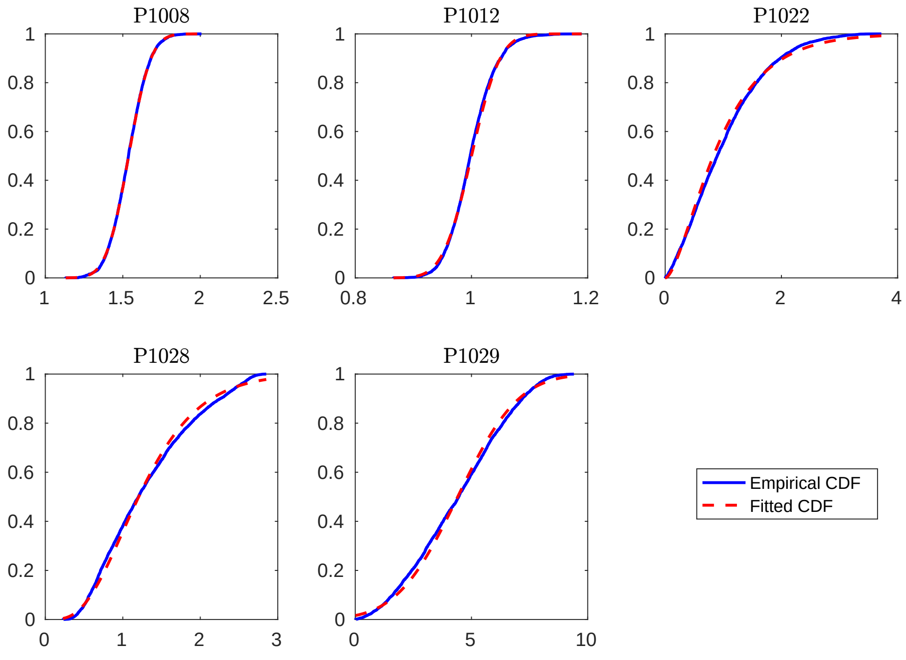


Fig. 6.11 Comparison of empirical CDFs and fitted CDFs.

observation data from this benchmark in our inverse UQ process. More informative data or `P1029`-sensitive benchmarks are required to better quantify its uncertainty.

6.6 Discussions

There are some limitations in the current application:

1. Model discrepancy is not considered in this study. In next Chapter, a different approach that can account for model discrepancy will be applied to the same problem.

2. SGSC surrogate model has demonstrated remarkable reduction in the simulation cost. However, stochastic spectral surrogate models suffer greatly from the "Curse of Dimensionality". SGSC can only be applied to problems with a relatively low dimension.

3. More advanced sparse grid methods can further reduce the computational cost (number of grid points), e.g. adaptive sparse grids.

4. The current study only used 8 out of 86 experiment tests from BFBT benchmark test assembly No.4. We have used some ad-hoc criteria to select these 8 tests. For instance, only observation data from high pressure and high power were used. All the tests with very low void fractions were abandoned. In future analysis more tests should be used.

5. Only void fraction data from upper elevations are used for inverse UQ because void fraction measurements at lower elevations are sometimes physically wrong (negative). However, those negative values are very close to zero. Given that the void fractions at those conditions may be very small, those slightly negative data can still be used.

Chapter 7

APPLICATION TO TRACE PHYSICAL MODEL PARAMETERS WITH GP

In this chapter, we use the same code and benchmark data with Chapter 6, but with GP metamodel. SGSC surrogate model can also greatly reduce the computational cost. However, unlike GP, it cannot be used to represent the model discrepancy during inverse UQ. In this work, we have greatly improved the application in the following aspects:

1. GP is used to construct metamodel for TRACE code, which requires even less TRACE runs than SGSC surrogate model used in Chapter 6. Furthermore, as shown in Chapter 4 GP metamodel provides MSE (variance) of its prediction which is essentially the "code uncertainty".

2. The work in Chapter 6 only used 8 experiment tests from BFBT benchmark test assembly No.4. In this work we will use all the 86 test cases.

3. Only void fraction data from upper elevations are used for inverse UQ in Section 6 because void fraction measurements at lower elevations are sometimes physically wrong (negative). In this work, we will use all the void fraction data considering that those negative void fraction measurements are very close to zero.

4. The work in Chapter 6 did not consider model discrepancy during inverse UQ. Therefore, the results are likely to be over-fitted to the selected test cases. In this work, we will describe the model discrepancy term with GP to avoid over-fitting, following the steps outlined in Chapter 2.

An unresolved issue for inverse UQ is "test source allocation (TSA)". TSA is the process to separate given experimental data for training (calibration) and testing (validation), as same data should not be used for both purposes. Very little previous research on Bayesian calibration dealt with TSA. Some researchers simply used random selection to separate observation data [110]. In Chapter 6, we separated tests by certain ad-hoc criteria, for example, tests with high pressure and high power are selected for inverse UQ. In another work [151], because the BFBT benchmark experiments are arranged according to the magnitude of power, pressure and mass flow rate, the authors simple picked every third test for inverse UQ and used the rest tests for validation.

A data partition methodology adapted from cross-validation was presented in [95]. The method aimed at separating legacy data for calibration and validation purposes. It considered all possible partitions and tried to find the optimal partition satisfying the following desiderata: (1) the model is sufficiently informed by the calibration tests, (2) the validation tests challenges the model as much as possible with respect to the QoIs. However, this method is extremely expensive. For an original data set of size N, the number of inverse (calibration) problems to solve is $2^N - 2$. For example, for 10 experiment tests, 1022 inverse problems need to be solved. This approach is apparently not practical to our application which has 86 test cases. Recently, a test selection methodology was developed [96] that involves an optimization framework for integrating calibration and validation data to make a prediction. In this approach, the TSA is motivated by uncertainty reduction in prediction. However, this method is design for a situation where the actual experiments have not been conducted yet, rather than selection among existing experiments.

In this chapter, we proposed an sequential approach for TSA. This algorithm includes three steps: (1) selecting an initial set for validation from all the tests; (2) selecting an initial set for inverse UQ after removing the initial validation set; (3) sequentially adding test cases for inverse UQ from the remaining tests. This algorithm guarantees that the tests used for validation have a maximum coverage of the test domain. Meanwhile, the tests used for inverse UQ have the lowest discrepancy which means that they explore the test space to the largest extent. The work presented in this chapter is also published in [151].

7.1 Problem Definition

In this study, $\mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta})$ is TRACE code. The outputs $\mathbf{y}^{\mathrm{M}}$ are void fractions. $\mathbf{x}$ stands for the four test conditions: ICs/BCs, including pressure, mass flow rate, power and inlet temperature. $\boldsymbol{\theta}$ represents the five uncertain physical model parameters, including `P1008`, `P1012`, `P1022`, `P1028` and `P1029`. Again we use BFBT assembly 4 void fraction data because it has high burnup which has 86 test cases. The version of TRACE we used in this chapter (version 5.0 Patch 4) in slightly different with what was used in Chapter 6 (version 5.0 Patch 3).

Figure 7.1 shows a comparison of void fraction from BFBT measurements (before and after correction) and TRACE simulations. All 86 test cases are included. Before data correction, the majority of the void fractions are under-predicted especially for VoidF1, VoidF2 and VoidF3. After data correction, the data points are more concentrated close to the diagonal line, indicating good agreements between BFBT and TRACE.

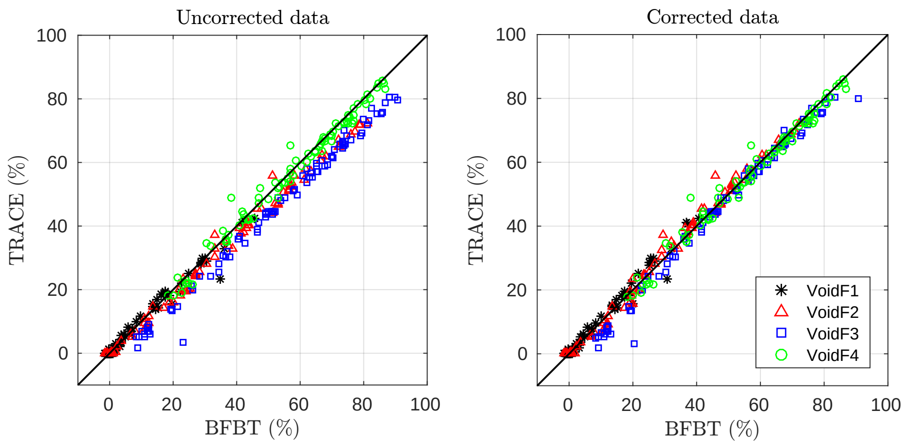


Fig. 7.1 Comparison of void fractions from BFBT and TRACE for all 86 tests of assembly 4.

It can be noticed that many of VoidF1 and VoidF2 values are very close to zero. In fact many of them are negative and were not considered in Chapter 6 because they are physically wrong. However, in this study we do not abandon these measurements considering that they are only slightly negative. There are also some measurement data that have substantially higher void fractions at lower elevations (e.g. VoidF3 > VoidF4). We decide to remove these non-physical observations. Finally, from Figure 1 (right) we can see some outliers, for instance, the notable VoidF3 by BFBT ($\sim 20\%$) for which TRACE simulation is much smaller ($\sim 4\%$). These outliers are removed because they have remarkable TRACE simulation errors compared with the majority. They are believed to be caused by measurements failures. Eventually, 8 tests are removed from the original 86 tests. The remaining 78 tests ($78 * 4 = 312$ void fraction observations) will be used in the following study.

7.1.1 Dimension reduction using sensitivity analysis

We used the sensitivity study results in Chapter 6 to identify the significant parameters for this problem. Table 7.1 shows the five parameters selected after dimension reduction. The nominal values for all these calibration parameters are 1.0 since they are multiplication factors. The prior ranges are chosen as $[0, 5]$ for all the parameters which will be used in design of computer experiments.

Table 7.1 Selected TRACE physical model parameters after sensitivity analysis

Parameter (multiplication factors)	Parameter	Uniform ranges	Nominal
Single phase liquid to wall HTC	`P1008`	[0.0, 5.0]	1.0
Subcooled boiling HTC	`P1012`	[0.0, 5.0]	1.0
Wall drag coefficient	`P1022`	[0.0, 5.0]	1.0
Interfacial drag (bubbly/slug Rod Bundle - Bestion) coefficient	`P1028`	[0.0, 5.0]	1.0
Interfacial drag (bubbly/slug Vessel) coefficient	`P1029`	[0.0, 5.0]	1.0

Again only main effects and total effects are calculated. Main effect represents the standalone influence of a certain input on the QoI, while total effect also accounts for the interaction of this input with the others. The fact that a certain input has close main and total effects means that this input has no interaction with others (e.g. `P1028`). If total effect is larger than the main effect, this input has interaction with others (e.g. `P1008` and `P1012`), the degree of which depends on the difference between main and total effects.

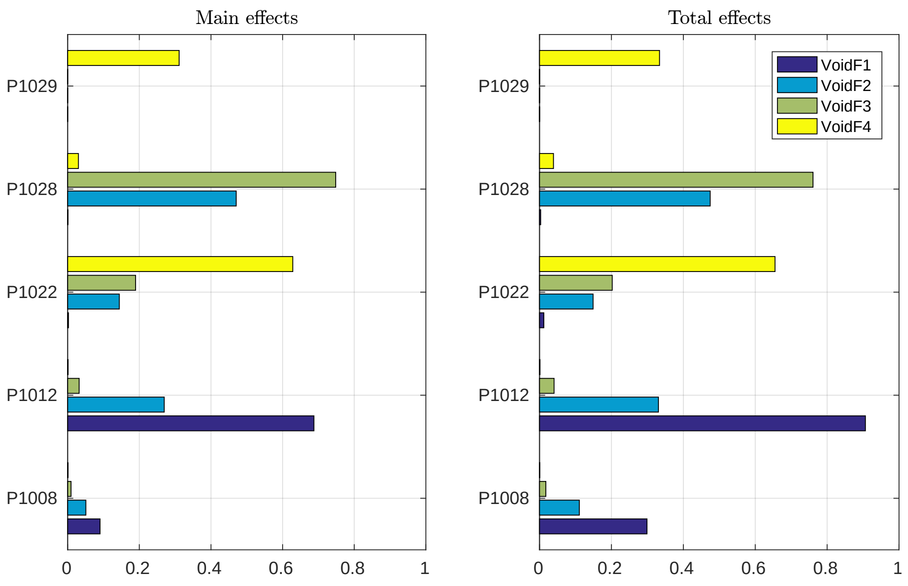


Fig. 7.2 Sobol' indices (main and total effects) for selected five physical model parameters.

7.1.2 Workflow for the investigated problem

The study in this chapter will follow the improved modular Bayesian approach presented in Section 2.3. The flowchart in Figure 2.6 consists of five major steps. In the following sections, each step will be discussed.

7.2 Test Source Allocation

In this section, a sequential approach for efficient TSA is developed (blocks connected by black arrows in Figure 2.6). This section starts with a brief introduction of discrepancy measures, which is later used as a measure of the degree of uniformity for the distribution of inverse UQ tests cases in the whole test domain. The following sub-sections describe the sequential approach for TSA, as well as two algorithms to select initial sets for validation and inverse UQ.

7.2.1 Discrepancy measure

In Chapter 4, we briefly mentioned low-discrepancy sequences for design of computer experiments. Low discrepancy sequences are deterministic designs constructed to uniformly fill the space. Various discrepancy measures can be used to judge the uniformity quality of the design, see discussions in [66]. Discrepancy measures based on L^2 norms can be analytically expressed which makes them the most popular in practice among others. For example, given a design $\mathbf{X}(n) = \left\{ x_k^{(i)}, i = 1,2,...,n, k = 1,2,...,d \right\}$ where n is the number of design points and d is the dimension of each design point, the centered L^2 discrepancy is calculated as:

$$D^2\left[\mathbf{X}(n)\right] = \left(\frac{13}{12}\right)^d - \frac{2}{n}\sum_{i=1}^{n}\prod_{k=1}^{d}\left(1+\frac{1}{2}\left|u_k^{(i)}-\frac{1}{2}\right|-\frac{1}{2}\left|u_k^{(i)}-\frac{1}{2}\right|^2\right) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\prod_{k=1}^{d}\left(1+\frac{1}{2}\left|u_k^{(i)}-\frac{1}{2}\right|+\frac{1}{2}\left|u_k^{(j)}-\frac{1}{2}\right|-\frac{1}{2}\left|u_k^{(i)}-u_k^{(j)}\right|\right) \tag{7.1}$$

Where $\left\{ u_k^{(i)}, i = 1,2,...,n, k = 1,2,...,d \right\}$ are the normalized values of $\mathbf{X}(n)$ in the interval $[0,1]$. A design sequence with a smaller centered L^2 discrepancy has a better coverage of the domain. Another recommended discrepancy measure is the wrap-around L^2 discrepancy defined in Equation 7.2, which allows to suppress bound effects [66] (by wrapping the unit cube for each dimension).

$$W^2\left[\mathbf{X}(n)\right] = \left(\frac{4}{3}\right)^d + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\prod_{k=1}^{d}\left(\frac{3}{2}-\left|u_k^{(i)}-u_k^{(j)}\right|\cdot\left(1-\left|u_k^{(i)}-u_k^{(j)}\right|\right)\right) \tag{7.2}$$

Besides their application in design of computer experiments, researchers have found other applications of low discrepancy sequences. For example, a "sequential validation design" was developed in [66] to select test points for the validation of metamodels. Suppose the metamodel was originally trained based on the sample set $\mathbf{X}_s$. Test points from a low discrepancy sequence $\mathbf{X}_f$ (e.g. Sobol, Halton, Hammersley, etc.) were selected one-by-one according to the criterion that among all the remaining points in $\mathbf{X}_f$, the selected point results in the minimal centered L^2 discrepancy after being added to $\mathbf{X}_s$. This design algorithm can avoid the possibility of too strong proximity between training sites and test sites, because it is capable of putting points in the unfilled zones of the training design.

7.2.2 A sequential approach for test source allocation

In the current work, we employ the similar idea with "sequential validation design" to separate the test cases for inverse UQ and validation. Given N_{test} experimental tests on the test domain $\mathbf{x}^{\text{test}}$, we would like to select N_{IUQ} tests for inverse UQ and the rest N_{VAL} tests to validate the updated model after inverse UQ. Denote the inverse UQ domain and validation domain as $\mathbf{x}^{\text{IUQ}}$ and $\mathbf{x}^{\text{VAL}}$ respectively. Then we have:

$$\mathbf{x}^{\text{test}} = \mathbf{x}^{\text{IUQ}} \cup \mathbf{x}^{\text{VAL}}, \qquad N_{\text{test}} = N_{\text{IUQ}} + N_{\text{VAL}}$$

Following the notations in Chapter 2, each of the input settings is a r-dimensional vector $\mathbf{x} = [x_1, x_2, ..., x_r]^\top$ representing r different design variables. In the current work $r = 4$. Figure 7.3 shows the workflow of the sequential approach for TSA. It includes the following key steps:

- Step 1: find initial set for validation:
  The initial set for validation is denoted as $\mathbf{x}^{\text{VAL,init}}$. The set $\mathbf{x}^{\text{VAL,init}}$ tests are selected from all the tests in $\mathbf{x}^{\text{test}}$. The selection criterion is that the validation experiments should have a full coverage of the test domain. The motivation and solution are explained later.

- Step 2: find initial set for inverse UQ:
  After removing the validation initial set from the test domain, the remaining tests are defined as $\mathbf{x}^{\text{rest}} = \mathbf{x}^{\text{test}} \backslash \mathbf{x}^{\text{VAL,init}}$. In this step we select initial set for inverse UQ from $\mathbf{x}^{\text{rest}}$. The selection criterion is that the tests in $\mathbf{x}^{\text{IUQ,init}}$ tend to be "far away" from other tests in the domain of interest. The motivation and solution are explained later.

- Step 3: sequentially add more tests for inverse UQ:
  Again we remove $\mathbf{x}^{\text{IUQ,init}}$ from $\mathbf{x}^{\text{rest}}$: $\mathbf{x}^{\text{rest}} = \mathbf{x}^{\text{rest}} \backslash \mathbf{x}^{\text{IUQ,init}}$. This step loops through all the remaining tests in $\mathbf{x}^{\text{rest}}$ to add one test each time to $\mathbf{x}^{\text{IUQ}}$. At the beginning, $\mathbf{x}^{\text{IUQ}} = \mathbf{x}^{\text{IUQ,init}}$. Then the algorithm find the test $\mathbf{x}^{(i^*)}$ from $\mathbf{x}^{\text{rest}}$ such that after being added to $\mathbf{x}^{\text{IUQ}}$, the minimum centered L^2 discrepancy $D^2\left[\mathbf{x}^{\text{IUQ}} \cup \mathbf{x}^{(i^*)}\right]$ or wrap-around L^2 discrepancy $W^2\left[\mathbf{x}^{\text{IUQ}} \cup \mathbf{x}^{(i^*)}\right]$ is achieved. Next the test $\mathbf{x}^{(i^*)}$ will be moved to the inverse UQ set: $\mathbf{x}^{\text{IUQ}} = \mathbf{x}^{\text{IUQ}} \cup \mathbf{x}^{(i^*)}, \mathbf{x}^{\text{rest}} = \mathbf{x}^{\text{rest}} \backslash \mathbf{x}^{(i^*)}$.

- Step 4: decision about terminating the sequential approach:
  This step decides whether the desirable number of tests for inverse UQ has been reached $N_{\text{IUQ}} = \lfloor N_{\text{test}} \cdot \alpha \rfloor$ or not. The round-down symbol $\lfloor N \rfloor$ means we take the largest integer that does not exceed N. In the current study we choose $\alpha = 25\%$ which means we use about 25% of all the tests for inverse UQ, and all the rest for validation $\mathbf{x}^{\text{VAL}} = \mathbf{x}^{\text{test}} \backslash \mathbf{x}^{\text{IUQ}}$. If the desirable number is reached, we proceed to perform inverse UQ with $\mathbf{x}^{\text{IUQ}}$. Otherwise, repeat step 3 to add another test for $\mathbf{x}^{\text{IUQ}}$.

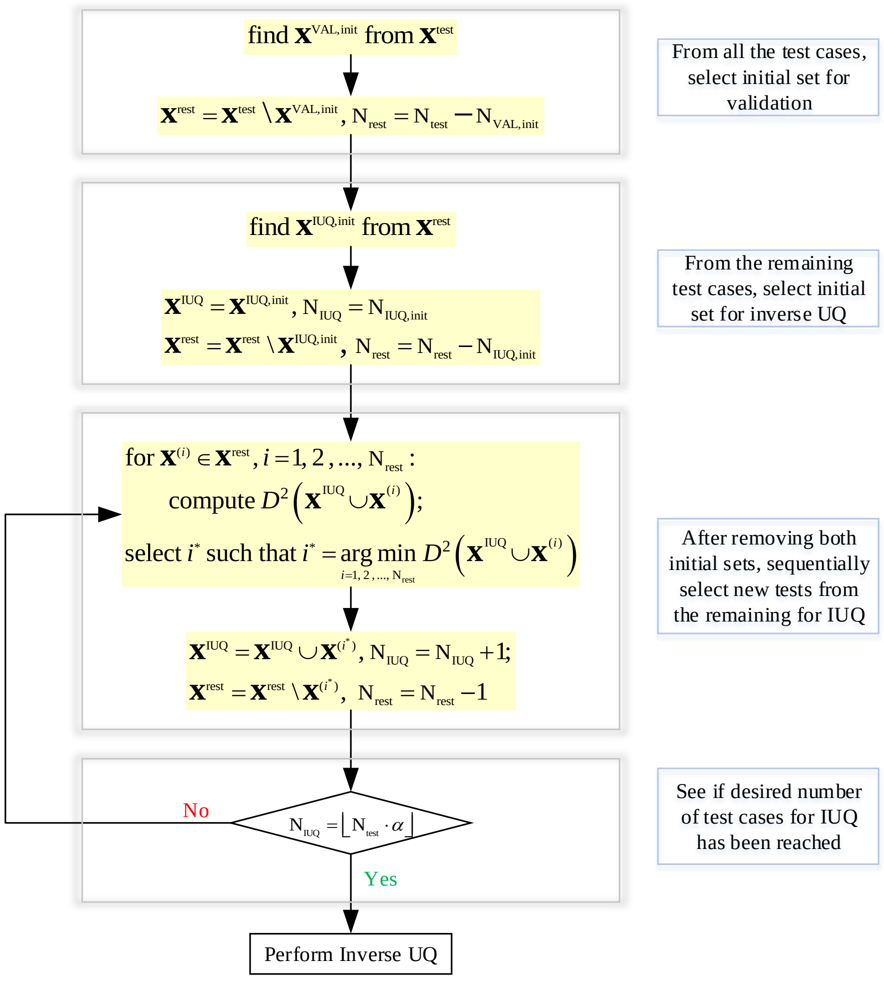

find $\mathbf{X}^{\text{VAL,init}}$ from $\mathbf{X}^{\text{test}}$
$\mathbf{X}^{\text{rest}} = \mathbf{X}^{\text{test}} \backslash \mathbf{X}^{\text{VAL,init}}$, $\text{N}_{\text{rest}} = \text{N}_{\text{test}} - \text{N}_{\text{VAL,init}}$
From all the test cases, select initial set for validation
find $\mathbf{X}^{\text{IUQ,init}}$ from $\mathbf{X}^{\text{rest}}$
$\mathbf{X}^{\text{IUQ}} = \mathbf{X}^{\text{IUQ,init}}$, $\text{N}_{\text{IUQ}} = \text{N}_{\text{IUQ,init}}$
$\mathbf{X}^{\text{rest}} = \mathbf{X}^{\text{rest}} \backslash \mathbf{X}^{\text{IUQ,init}}$, $\text{N}_{\text{rest}} = \text{N}_{\text{rest}} - \text{N}_{\text{IUQ,init}}$
From the remaining test cases, select initial set for inverse UQ
for $\mathbf{X}^{(i)} \in \mathbf{X}^{\text{rest}}, i = 1, 2, \ldots, \text{N}_{\text{rest}}$:
compute $D^2\left(\mathbf{X}^{\text{IUQ}} \cup \mathbf{X}^{(i)}\right)$;
select i^* such that $i^* = \arg\min_{i=1,2,\ldots,\text{N}_{\text{rest}}} D^2\left(\mathbf{X}^{\text{IUQ}} \cup \mathbf{X}^{(i)}\right)$
$\mathbf{X}^{\text{IUQ}} = \mathbf{X}^{\text{IUQ}} \cup \mathbf{X}^{(i^*)}$, $\text{N}_{\text{IUQ}} = \text{N}_{\text{IUQ}} + 1$;
$\mathbf{X}^{\text{rest}} = \mathbf{X}^{\text{rest}} \backslash \mathbf{X}^{(i^*)}$, $\text{N}_{\text{rest}} = \text{N}_{\text{rest}} - 1$
After removing both initial sets, sequentially select new tests from the remaining for IUQ
No
$\text{N}_{\text{IUQ}} = \lfloor \text{N}_{\text{test}} \cdot \alpha \rfloor$
Yes
See if desired number of test cases for IUQ has been reached
Perform Inverse UQ

Fig. 7.3 The sequential approach for test source allocation.

Apparently, following the steps outlined in Figure 7.3, $\mathbf{x}^{\text{IUQ,init}}$ and $\mathbf{x}^{\text{VAL,init}}$ will be subsets of and $\mathbf{x}^{\text{VAL}}$, respectively. The step 3 of the proposed sequential approach will select the test "furthest away" from the existing tests in $\mathbf{x}^{\text{IUQ}}$. By choosing the test whose input setting is in the unfilled zone of the existing test domain, we aim at extracting the most information for $\boldsymbol{\theta}^{\text{Posterior}}$ using only a relatively small number of N_{IUQ}.

7.2.3 Method to select initial set for validation

In our improved modular Bayesian approach outlined in Figure 2.6, the computer code $\mathbf{y}^{\mathrm{M}}(\mathbf{x}, \boldsymbol{\theta})$ is first executed at the input settings of all the tests $\mathbf{x}^{\mathrm{test}}$, with the calibration parameters fixed at nominal values or prior mean values $\boldsymbol{\theta}^0$. The resulting simulations are denoted as $\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{test}}, \boldsymbol{\theta}^0)$. Then $\mathbf{x}^{\mathrm{VAL}}$ and $\left\{\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}}) - \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^0)\right\}$ are used as training inputs and output respectively to fit a GP emulator called `GPbias`. Evaluating `GPbias` at $\mathbf{x}^{\mathrm{IUQ}}$ results in an estimation of the model discrepancy term $\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)$, which will enter the likelihood function during MCMC sampling. Such a treatment of model discrepancy provide the following implications for TSA:

- Because generally a larger training sample size results in a more accurate GP model, N_{VAL} should be relatively larger than N_{IUQ}. That's why we have picked $\alpha = 25\%$ in Step 4 of the sequential approach for TSA. In this way, about 75% of the measurement data will be used for validation. Furthermore, a relatively smaller number of tests for inverse UQ leads to smaller computational cost of MCMC sampling.

- It was also demonstrated in Chapter 4 that GP emulator is very accurate for interpolation but can cause large errors when used for extrapolation. This fact implies that since `GPbias` is trained with $\mathbf{x}^{\mathrm{VAL}}$ and evaluated at $\mathbf{x}^{\mathrm{IUQ}}$, the inverse UQ domain $\mathbf{x}^{\mathrm{IUQ}}$ should be encompassed by the validation domain $\mathbf{x}^{\mathrm{VAL}}$ to avoid extrapolation. Note that this does not mean that $\mathbf{x}^{\mathrm{IUQ}}$ should be a subset of $\mathbf{x}^{\mathrm{VAL}}$. Each test must be either in $\mathbf{x}^{\mathrm{IUQ}}$ or in $\mathbf{x}^{\mathrm{VAL}}$. If $\mathbf{x}^{\mathrm{IUQ}}$ is a subset of $\mathbf{x}^{\mathrm{VAL}}$, $\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)$ will have zero mean and zero variance, given the fact that GP is an interpolator.

Algorithm 1: select experimental tests for initial validation set $\mathbf{x}^{\mathrm{VAL,init}}$

```
1. Computer the volume of the convex hull $\Omega(\mathbf{x}^{\mathrm{test}})$ defined by all the tests $\mathbf{x}^{\mathrm{test}}$: $\mathrm{Volume}[\Omega(\mathbf{x}^{\mathrm{test}})]$.
2. Choose starting tests for $\mathbf{u}$, which is denoted as $\mathbf{u}^{\mathrm{start}}$.
       $\mathbf{u} = \mathbf{u}^{\mathrm{start}}$,   $\mathrm{N}_{\mathbf{u}} = \mathrm{N}_{\mathbf{u},\mathrm{start}}$,   $\mathbf{u}^{\mathrm{rest}} = \mathbf{x}^{\mathrm{test}} \backslash \mathbf{u}^{\mathrm{start}}$,   $\mathrm{N}_{\mathbf{u},\mathrm{rest}} = \mathrm{N}_{\mathrm{test}} - \mathrm{N}_{\mathbf{u},\mathrm{start}}$
3. while $\mathrm{N}_{\mathbf{u}} < \mathrm{N}_{\mathrm{test}}$
       for $\mathbf{x}^{(\mathrm{k})} \in \mathbf{u}^{\mathrm{rest}}$, $k = 1,2,\dots,\mathrm{N}_{\mathbf{u},\mathrm{rest}}$, compute:
           $\eta_C\left[\Omega\left(\mathbf{u} \cup \mathbf{x}^{(\mathrm{k})}\right)\right] = \frac{\mathrm{Volume}\left[\Omega\left(\mathbf{u} \cup \mathbf{x}^{(\mathrm{k})}\right)\right]}{\mathrm{Volume}[\Omega(\mathbf{x}^{\mathrm{test}})]}$
       end
       select $k^* = \underset{k}{\mathrm{argmax}}\ \eta_C\left[\Omega\left(\mathbf{u} \cup \mathbf{x}^{(\mathrm{k})}\right)\right]$;
           $\mathbf{u} = \mathbf{u} \cup \mathbf{x}^{(k^*)}$,   $\mathbf{u}^{\mathrm{rest}} = \mathbf{u}^{\mathrm{rest}} \backslash \mathbf{x}^{(k^*)}$,   $\mathrm{N}_{\mathbf{u}} = \mathrm{N}_{\mathbf{u}} + 1$,   $\mathrm{N}_{\mathbf{u},\mathrm{rest}} = \mathrm{N}_{\mathbf{u},\mathrm{rest}} - 1$
   end
4. Eventually, all the tests in $\mathbf{x}^{\mathrm{test}}$ are re-ordered in $\mathbf{u}$.
```

Fig. 7.4 Algorithm to select experimental tests for initial validation set $\mathbf{x}^{\mathrm{VAL,\ init}}$

The selection of $\mathbf{x}^{\text{VAL,init}}$ can be guided by the above two implications, which states that validation tests $\mathbf{x}^{\text{VAL}}$ should have a full coverage of the test domain $\mathbf{x}^{\text{test}}$. The term "coverage" refers to the extent to which $\mathbf{x}^{\text{VAL}}$ explore the test domain (especially the boundaries) of the selected benchmark data. The estimation of the model discrepancy term $\delta\left(\mathbf{x}^{\text{IUQ}}\right)$ will be limited if the tests with which `GPbias` is trained have insufficient coverage of the test domain. The sequential approach for TSA determines the coverage of the test domain based on convex hull. Convex hull, also called convex envelope, is the smallest convex domain that envelops all the physical experiments. The coverage η_C is calculated using the following equation:

$$\eta_C = \frac{\text{Volume}\left[\Omega\left(\mathbf{x}^{\text{VAL}}\right)\right]}{\text{Volume}\left[\Omega\left(\mathbf{x}^{\text{test}}\right)\right]}$$

Where $\Omega\left(\mathbf{x}^{\text{VAL}}\right)$ is the convex hull of the domain defined by $\mathbf{x}^{\text{VAL}}$, $\Omega\left(\mathbf{x}^{\text{test}}\right)$ is the convex hull defined by $\mathbf{x}^{\text{test}}$. Function Volume$[\cdot]$ calculates the volume of convex hull.

The procedure to select $\mathbf{x}^{\text{VAL,init}}$ is shown in Figure 7.4. This algorithm tries to re-order $\mathbf{x}^{\text{test}}$ to $\mathbf{u}$. The major steps are briefly described below:

1. Firstly, the volume of the convex hull $\Omega\left(\mathbf{x}^{\text{test}}\right)$ is calculated;

2. Secondly, choose starting tests for $\mathbf{u}$, because at least $(r+1)$ tests are required to calculate the volume of a convex hull of dimension r. Such starting tests can be those whose input settings contain the minimum or maximum values of each design variable. Denote these starting tests as $\mathbf{u}^{\text{start}}$.

3. Thirdly, loop through all the remaining tests in $\mathbf{u}^{\text{rest}}$, select the one that maximize the coverage ratio if it is added to the current $\mathbf{u}$. Repeat this process until there is no more test in $\mathbf{u}^{\text{rest}}$.

4. Eventually, step 3 results in a set $\mathbf{u}$ which has the same number of tests with $\mathbf{x}^{\text{test}}$. However, these tests are reordered in $\mathbf{u}$ such that for any number $(N_{\text{u, start}}+1) \leq n \leq N_{\text{test}}$, the first n tests in $\mathbf{u}$ has the largest coverage η_C of the whole test domain among all the other possible combinations. $\mathbf{x}^{\text{VAL,init}}$ will include the first $N_{\text{VAL,init}}$ tests in $\mathbf{u}$ such that the coverage ratio becomes 1.0.

7.2.4 Method to select initial set for inverse UQ

The procedure to select tests for $\mathbf{x}^{\text{IUQ,init}}$ is shown in Figure 7.5 and briefly described below:

1. Firstly, remove $\mathbf{x}^{\text{VAL,init}}$ from $\mathbf{x}^{\text{test}}$: $\mathbf{x}^{\text{rest}} = \mathbf{x}^{\text{test}} \backslash \mathbf{x}^{\text{VAL,init}}$, and $\mathbf{x}^{\text{IUQ,init}}$ will be selected from $\mathbf{x}^{\text{rest}}$.

2. Secondly, select a desirable number of tests $N_{\text{IUQ,init}}$ for $\mathbf{x}^{\text{IUQ,init}}$. For example, $N_{\text{IUQ,init}} = \lfloor N_{\text{test}} \cdot \beta \rfloor$ where β is chosen by the user, e.g. $\beta = 0.05$.

3. Thirdly, starting at each of the tests in the test domain $\left\{\mathbf{x}^{(i)} \in \mathbf{x}^{\text{rest}}, i = 1, 2, \ldots, N_{\text{rest}}\right\}$, follow the step 3 in Algorithm 2 to select the rest $(N_{\text{IUQ,init}} - 1)$ tests for $\mathbf{x}^{\text{IUQ,init,(i)}}$.

4. Eventually we will have N_{rest} different sets $\left\{\mathbf{x}^{\text{IUQ,init,(i)}}, i = 1, 2, \ldots, N_{\text{rest}}\right\}$, each set includes $N_{\text{IUQ,init}}$ experimental tests and the index (i) means the starting test. All together there are $N_{\text{rest}} \cdot N_{\text{IUQ,init}}$ counts of single test appearances, of which many tests will have multiple appearances. The fact that a certain test appears more frequently means that it is likely to be in an unfilled region in the test domain. We then rank the counts of appearance and select the top $N_{\text{IUQ,init}}$ tests to form the initial inverse UQ set $\mathbf{x}^{\text{IUQ,init}}$.

Algorithm 2: select experimental tests for initial inverse UQ set $\mathbf{x}^{\text{IUQ,init}}$

```
1. Remove $\mathbf{x}^{\text{VAL,init}}$ from $\mathbf{x}^{\text{test}}$: $\mathbf{x}^{\text{rest}} = \mathbf{x}^{\text{test}} \backslash \mathbf{x}^{\text{VAL,init}}$.
2. Decide the size for $\mathbf{x}^{\text{IUQ,init}}$: $\text{N}_{\text{IUQ,init}} = \lfloor \text{N}_{\text{test}} * \beta \rfloor$ where $\beta$ is chosen by the user (e.g. 0.05).
3. for $\mathbf{x}^{(\text{i})} \in \mathbf{x}^{\text{rest}},\ i = 1,2,\ldots,\text{N}_{\text{rest}}$:
           $\mathbf{x}^{\text{IUQ,init,(i)}} = \mathbf{x}^{(\text{i})}$;
           $n = 1$;
           while $n < \text{N}_{\text{IUQ,init}}$
                 for $\mathbf{x}^{(\text{k})} \in \left(\mathbf{x}^{\text{rest}} \backslash \mathbf{x}^{\text{IUQ,init,(i)}}\right),\ k = 1,2,\ldots,(\text{N}_{\text{rest}} - n)$
                       compute $D^2\left[\mathbf{x}^{\text{IUQ,init,(i)}} \cup \mathbf{x}^{(\text{k})}\right]$ or $W^2\left[\mathbf{x}^{\text{IUQ,init,(i)}} \cup \mathbf{x}^{(\text{k})}\right]$;
                 end
                 select $k^* = \underset{k}{\operatorname{argmin}}\, D^2\left[\mathbf{x}^{\text{IUQ,init,(i)}} \cup \mathbf{x}^{(\text{k})}\right]$ or $k^* = \underset{k}{\operatorname{argmin}}\, W^2\left[\mathbf{x}^{\text{IUQ,init,(i)}} \cup \mathbf{x}^{(\text{k})}\right]$;
                 $\mathbf{x}^{\text{IUQ,init,(i)}} = \mathbf{x}^{\text{IUQ,init,(i)}} \cup \mathbf{x}^{(k^*)}$;
                 $n = n + 1$;
           end
   end
4. Step 3 results in $\text{N}_{\text{rest}}$ different sets $\{\mathbf{x}^{\text{IUQ,init,(i)}}, i = 1,2,\ldots,\text{N}_{\text{rest}}\}$, each set includes $\text{N}_{\text{IUQ,init}}$ experimental tests. There
   are $\text{N}_{\text{rest}} \times \text{N}_{\text{IUQ,init}}$ counts of single test appearances. Select the top $\text{N}_{\text{IUQ,init}}$ tests which have the most counts to form
   the initial set $\mathbf{x}^{\text{IUQ,init}}$.
```

Fig. 7.5 Algorithm to select experimental tests for initial inverse UQ set $\mathbf{x}^{\text{IUQ, init}}$

The Algorithm 2 in Figure 7.5 can efficiently identify those tests that are "far away" from other tests. Such tests have a higher possibility to be selected by the sequential approach in Figure 7.3. However, putting them in the initial set $\mathbf{x}^{\text{IUQ,init}}$ will speed up the TSA process.

7.2.5 TSA results for the BFBT problem

Figure 7.6 shows the increasing of the coverage ratio with the number of tests in $\mathbf{u}$ for the current problem. The fact that x-axis starts at 6 means that there are 6 tests in $\mathbf{u}^{\text{start}}$ which include all the lower and upper bounds of the four design variables. We found that a coverage ratio of 1.0 is reached with the first 25 tests in $\mathbf{u}$. Therefore, $\mathbf{x}^{\text{VAL,init}}$ takes all these 25 tests and $\mathbf{x}^{\text{IUQ,init}}$ will be searched for among all the remaining tests.

In this study, we chose $\beta = 0.05$, which means $\mathbf{x}^{\text{IUQ, init}}$ contains $\lfloor 78 \times 0.05 \rfloor = 3$ tests. Furthermore, we use a value of 0.25 for α so that $\mathbf{x}^{\text{IUQ}}$ consists of $\lfloor 78 \times 0.25 \rfloor = 19$ tests. Figure 7.7 shows the values for the four design variables: pressure, mass flow rate, power and inlet temperature. It is obvious that the tests for inverse UQ distribute evenly in the test domain. In fact, they have the lowest wrap-around L^2 discrepancy among all the possible combinations of the same number of tests. Figure 7.8 shows the void fractions for the two separated sets.

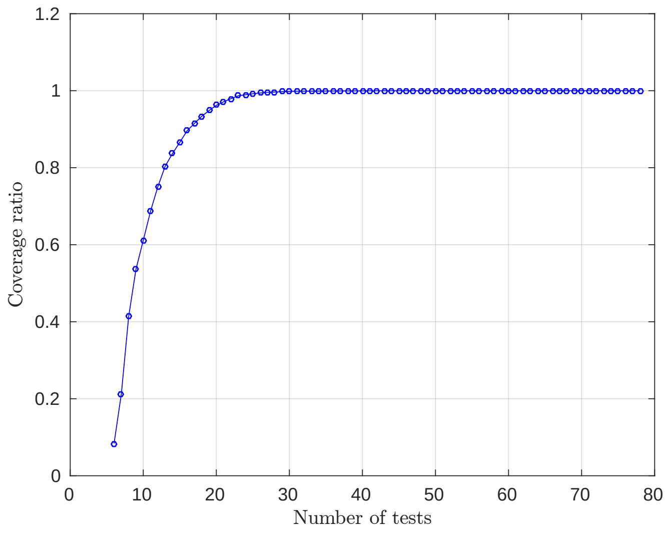


Fig. 7.6 Relation of the coverage ratio with the number of tests in $\mathbf{u}$.

7.3 Modeling the TRACE model discrepancy

This section presents the results for the modeling of TRACE model discrepancy $\delta(\mathbf{x})$ (blocks connected by green arrows in Figure 2.6). The TRACE code is first evaluated at the input settings of all the tests $\mathbf{x}^{\text{test}}$, with the calibration parameters fixed at nominal values or prior mean values $\boldsymbol{\theta}^0$. The resulting model simulations $\mathbf{y}^{\text{M}}(\mathbf{x}^{\text{test}}, \boldsymbol{\theta}^0)$ can be compared with $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{test}})$ to get TRACE prediction errors. Figure 7.9 shows the TRACE prediction errors for inverse UQ tests $\left\{\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{IUQ}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^0)\right\}$ and validation tests $\left\{\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^0)\right\}$. The x-axis is the test ID ranges from 1 to 86. It can be seen that TRACE tends to under-predict VoidF3 and VoidF4, while over-predict VoidF1.

Next, $\mathbf{x}^{\text{VAL}}$ and the corresponding TRACE prediction errors $\left\{\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^0)\right\}$ are used as training inputs and outputs for the GP emulator `GPbias`. At any $\mathbf{x}^*$, `GPbias` will provide a prediction of the model discrepancy, which is a Gaussian distribution. Evaluating

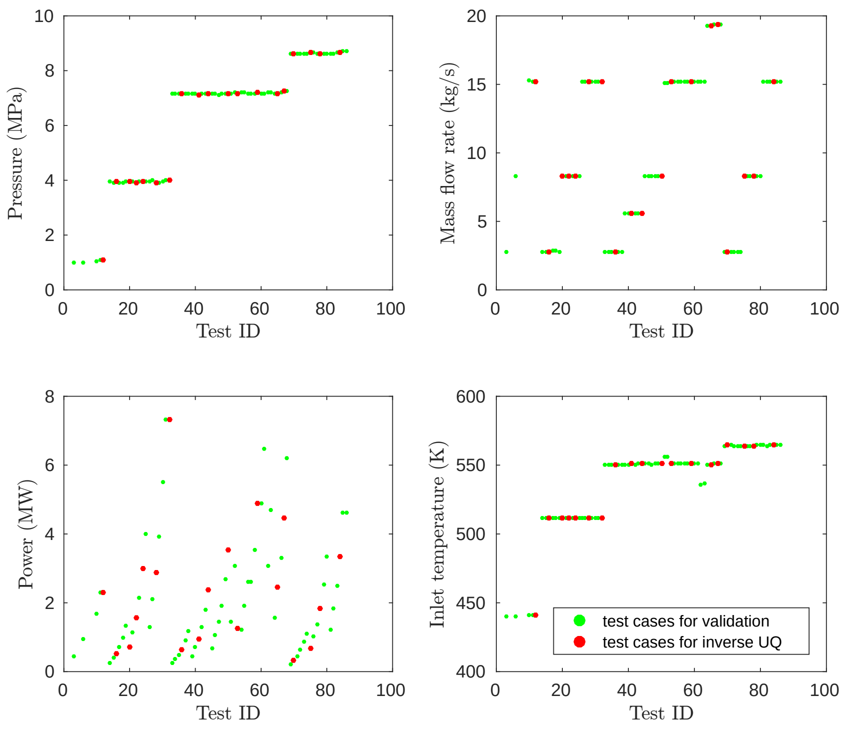


Fig. 7.7 Design variables distribution for $\mathbf{x}^{\mathrm{IUQ}}$ and $\mathbf{x}^{\mathrm{VAL}}$.

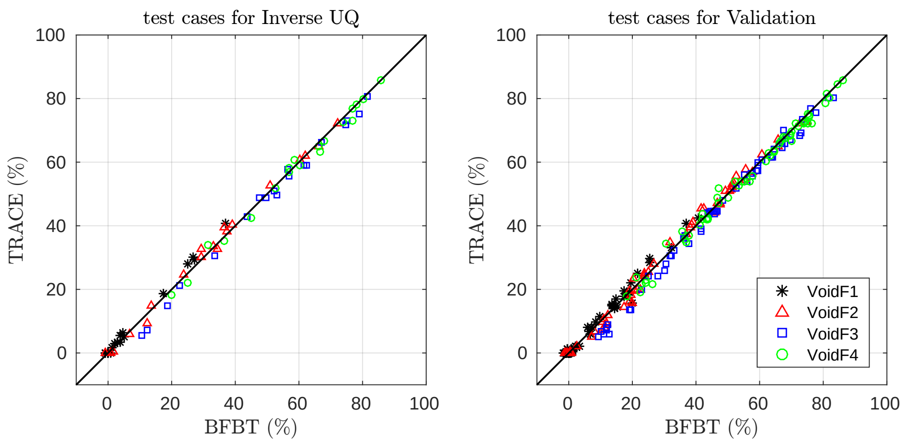


Fig. 7.8 Void fractions for $\mathbf{y}^{\mathrm{E}}\left(\mathbf{x}^{\mathrm{IUQ}}\right)$ and $\mathbf{y}^{\mathrm{E}}\left(\mathbf{x}^{\mathrm{VAL}}\right)$.

`GPbias` at $\mathbf{x}^{\mathrm{IUQ}}$ results in $\delta(\mathbf{x}^{\mathrm{IUQ}})$ and $\boldsymbol{\Sigma}_{\mathrm{bias}}$, where the former is a mean vector that contains the "expected prediction error" of TRACE at $\mathbf{x}^{\mathrm{IUQ}}$, and the latter is a matrix whose diagonal[1] entries are the variances of the mean vector. The mean vector $\delta(\mathbf{x}^{\mathrm{IUQ}})$ and covariance matrix $\boldsymbol{\Sigma}_{\mathrm{bias}}$ will enter the likelihood function (Equation 2.5) during MCMC sampling.

[1]In this work, $\boldsymbol{\Sigma}_{\mathrm{bias}}$ is a diagonal matrix because we do not have enough information for the correlation of the prediction errors in different QoIs. But the proposed inverse UQ approach can readily be extended to incorporate such correlations once available.

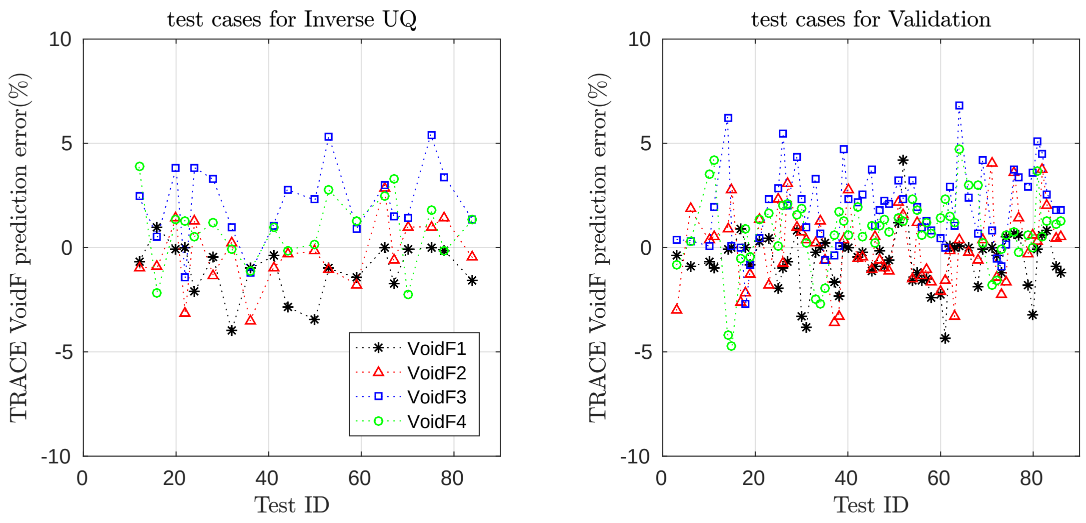


Fig. 7.9 The error in TRACE void fraction simulation for inverse UQ and validation test sets.

Figure 7.10 shows the comparison of the "actual TRACE prediction errors" $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}}) - \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{IUQ}}, \boldsymbol{\theta}^0)$ and the "expected or interpolated TRACE prediction errors" $\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)$. Also note that the standard deviations $\sqrt{\mathrm{MSE}\left[\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)\right]}$ are plotted as errorbar. Recall the likelihood:

$$\frac{\exp\left[-\frac{1}{2}\left[\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}}) - \mathbf{y}^{\mathrm{M}} - \delta(\mathbf{x}^{\mathrm{IUQ}})\right]^{\top}\left[\boldsymbol{\Sigma}_{\mathrm{bias}} + \boldsymbol{\Sigma}_{\mathrm{code}} + \boldsymbol{\Sigma}_{\mathrm{exp}}\right]^{-1}\left[\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{IUQ}}) - \mathbf{y}^{\mathrm{M}} - \delta(\mathbf{x}^{\mathrm{IUQ}})\right]\right]}{\sqrt{\left|\boldsymbol{\Sigma}_{\mathrm{bias}} + \boldsymbol{\Sigma}_{\mathrm{code}} + \boldsymbol{\Sigma}_{\mathrm{exp}}\right|}}$$

There are three possibilities:

- $\delta(\mathbf{x}^{\mathrm{IUQ}}) \approx \mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}}$, $\sqrt{\mathrm{MSE}\left[\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)\right]} \approx 0$. Nothing enters the likelihood function. This inverse UQ test is "not informative". For example, test ID 32 for VoidF3.;
- $\delta(\mathbf{x}^{\mathrm{IUQ}}) \approx \mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}}$, $\sqrt{\mathrm{MSE}\left[\delta\left(\mathbf{x}^{\mathrm{IUQ}}\right)\right]} \not\approx 0$. The standard deviation provides some information for the likelihood function. This inverse UQ test is "informative" For example, test ID 70 for VoidF2.;
- $\delta(\mathbf{x}^{\mathrm{IUQ}}) \not\approx \mathbf{y}^{\mathrm{E}} - \mathbf{y}^{\mathrm{M}}$. This inverse UQ test is "very informative". This is true for most inverse UQ tests.

7.4 Building and Validating GP Metamodel for TRACE

In this section, another GP emulator called `GPcode` is built for TRACE (blocks connected by purple arrows in Figure 2.6). Even though TRACE simulation for the BFBT benchmark in the present study is not very expensive (each TRACE simulation takes around 41 seconds), we

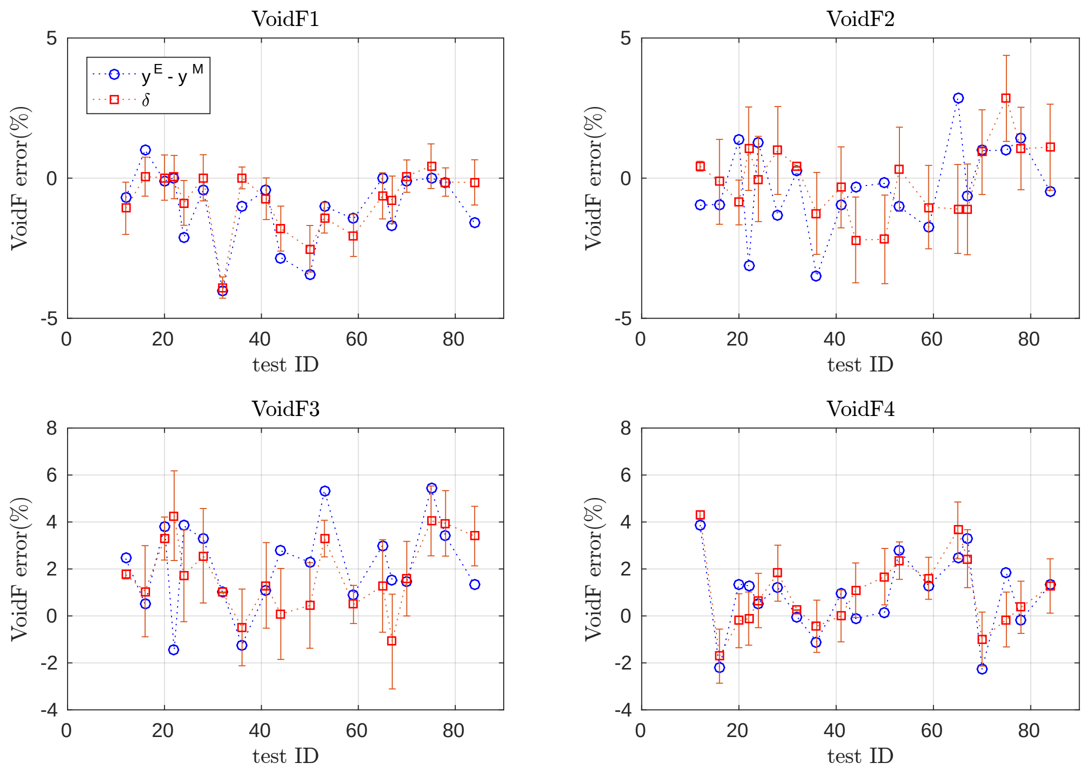


Fig. 7.10 Results of `GPbias` evaluated at $\mathbf{x}^{\text{IUQ}}$.

use it as a placeholder for more computationally prohibitive codes. For those expensive codes there will only be a limited number of runs (a few hundred) available for follow-on analysis. Moreover, 50,000 MCMC samples require the same number of TRACE full model evaluations, which could still take around 24 days with a single processor, making the application of GP emulator necessary.

7.4.1 Build the GP metamodel

The major difference of `GPcode` with `GPbias` is shown in Table 7.2. `GPbias` uses $\mathbf{x}^{\text{VAL}}$ as inputs and $\{\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^0)\}$ as outputs for training, while `GPcode` uses $(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^{\text{Prior}})$ as inputs and $\mathbf{y}^{\text{M}}(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^{\text{Prior}})$ as outputs because it is intended to serve as a surrogate model for TRACE. The prior $\boldsymbol{\theta}^{\text{Prior}}$ are uniform distributions over the ranges in Table 7.1. Such non-informative priors are used to reflect our ignorance about $\boldsymbol{\theta}$. The prior ranges are very important since during MCMC sampling, any trial walk outside the prior ranges will have a zero acceptance probability, meaning that posterior samples will never fall beyond the prior ranges. If the prior ranges are too limited and do not include the "true values" of $\boldsymbol{\theta}$, inverse UQ can never converge at these "true values".

Table 7.2 Training inputs and outputs for `GPbias` and `GPcode`

	`GPbias`	`GPcode`
Training inputs	$\mathbf{x}^{\text{VAL}}$	$\left(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^{\text{Prior}}\right)$
Training outputs	$\mathbf{y}^{\text{E}}\left(\mathbf{x}^{\text{VAL}}\right) - \mathbf{y}^{\text{M}}\left(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^{0}\right)$	$\mathbf{y}^{\text{M}}\left(\mathbf{x}^{\text{IUQ}}, \boldsymbol{\theta}^{\text{Prior}}\right)$

Another notable difference is that, `GPbias` only uses existing values of $\mathbf{x}^{\text{VAL}}$ and does not perform a design of computer experiments at other $\mathbf{x}$ values where no measurement data exist. However, `GPcode` needs to be built with an experimental design of $\boldsymbol{\theta}$ following the distributions of $\boldsymbol{\theta}^{\text{Prior}}$, while keeping $\mathbf{x}^{\text{IUQ}}$ fixed. For every test in $\mathbf{x}^{\text{IUQ}}$, we generated N_{design} design samples of $\boldsymbol{\theta}$ using maximin LHS. At this moment we do not know how many design samples are enough to construct an accurate `GPcode` metamodel for TRACE, we tried N_{design} = 3, 6, 9, 12, 15 and 20. Therefore, for every N_{design} value, `GPcode` is trained with $N_{\text{IUQ}} \times N_{\text{design}}$ samples. MLE is used for estimation of the GP hyperparameters.

7.4.2 Validate the GP metamodel

In this work, to evaluate the accuracy of `GPcode` we calculate the predictivity coefficients Q_2 and LOOCV errors. Table 7.3 shows the convergence of these two indicators with different N_{design} values. Figure 7.11 and 7.12 visualize the results in Table 7.3. It is apparent that with the increasing of N_{design}, Q_2 values quickly converge to 1.0 for all four QoIs, and LOOCV errors decrease towards zeros. In Section 2.3 we have mentioned that in literature a metamodel with Q_2 value above 0.7 is often considered as a satisfactory approximation of the full model. In this work, we used a more stringent criterion and require the Q_2 values to be above 0.95. Eventually we pick $N_{\text{design}} = 20$ to build the `GPcode` emulator. It can readily be used to replace TRACE during MCMC sampling since its accuracy has been proven.

Table 7.3 Predictivity coefficients and LOOCV errors for each design

N_{design}	Predictivity coefficient				LOOCV error			
	VoidF1	VoidF2	VoidF3	VoidF4	VoidF1	VoidF2	VoidF3	VoidF4
3	0.0636	0.0456	0.0077	0.0870	2325.10	11159.00	14174.00	2797.40
6	0.6228	0.9482	0.9017	0.9443	67.41	26.84	51.71	20.05
9	0.9091	0.9716	0.9814	0.9829	12.78	15.14	9.31	6.02
12	0.9008	0.9564	0.9908	0.9978	13.07	23.89	4.59	0.77
15	0.9494	0.9670	0.9953	0.9971	6.92	17.77	2.34	1.01
20	0.9698	0.9841	0.9930	0.9959	4.26	8.58	3.50	1.43

The increase of Q_2 values and decrease of LOOCV errors are expected to be monotonic. However, the results in Figure 7.11 and 7.12 show that the evolution of these two indicators are not monotonic. This is caused by the randomness of the design of computer experiments. Our conclusion will not be affected as long as the overall trend is consistent with expected monotonic behavior.

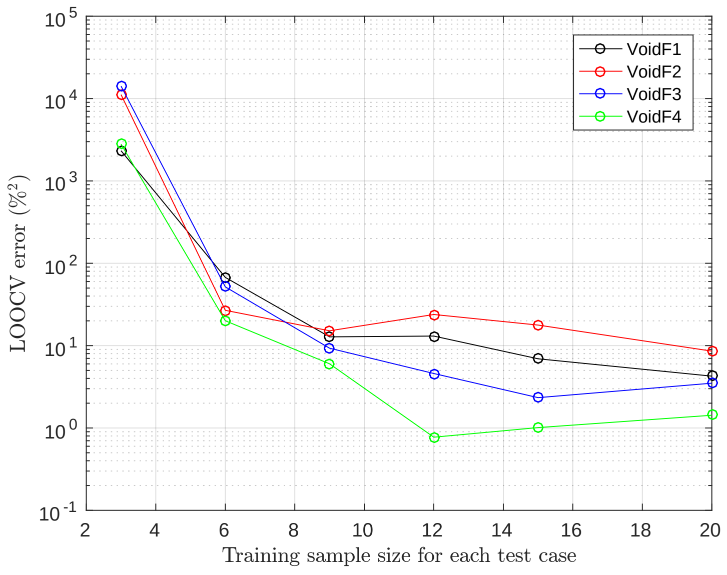


Fig. 7.11 Convergence of LOOCV errors.

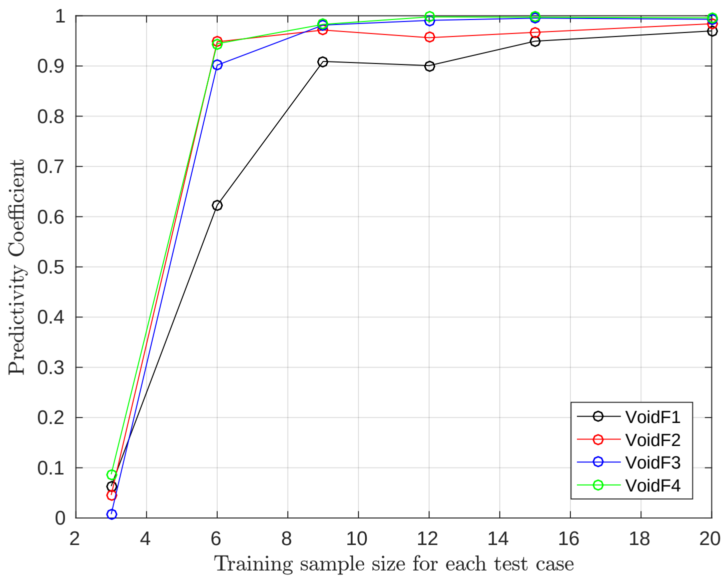


Fig. 7.12 Convergence of predictivity coefficients.

7.5 Results for Posteriors by MCMC Sampling

In this section, we present the results for posteriors by MCMC sampling (blocks connected by red arrows in Figure 2.6). The measurement error is assumed to be 2% relative to BFBT void fraction data, which is needed for the variance term in the posterior. Such a value is reported in BFBT benchmark specification [98]. Note that the benchmark only provides measurement noise related to X-ray CT scanner (VoidF4). We assign the same experimental error for X-ray densitometer (VoidF1, VoidF2 and VoidF3) as there is no better choice.

We used the adaptive MCMC sampling approaches described in Section 2.4. 50,000 samples were generated using `GPcode`. It took about 37 core-minutes using GP metamodel with a current generation Intel CPU, which would otherwise take about 24 core-days using TRACE full model with the same processor. The first 10,000 samples were discarded as burn-in and then every 10th sample were kept from the remainder for thinning of the chain, leaving us with 4000 samples. Thinning was performed to reduce auto-correlation among the samples. We also generated another MCMC chain without considering the model discrepancy, while keeping everything else the same. In that case, the likelihood function has a form shown in Equation 2.7.

Table 7.4 Posterior statistical moments

Parameter	With model discrepancy			Without model discrepancy		
	Mean	STD	Mode	Mean	STD	Mode
`P1008`	0.6162	0.2113	0.4967	1.5275	0.1923	1.3651
`P1012`	1.2358	0.0890	1.0559	1.0844	0.0592	1.0380
`P1022`	1.4110	0.1833	1.4096	0.2452	0.1153	0.2600
`P1028`	1.3385	0.1155	1.2044	1.4746	0.0414	1.4300
`P1029`	1.2340	0.3453	1.0675	0.4321	0.0833	0.2700

When inverse UQ is performed without considering the model discrepancy, all the actions indicated by green arrows in Figure 2.6 will be gone. The posterior $\boldsymbol{\theta}^{\text{Posterior}}$ is expected to be over-fitted to $\mathbf{y}^{\text{E}}\left(\mathbf{x}^{\text{IUQ}}\right)$ and we want to prove that with the current treatment of model discrepancy in Section 2.3, such over-fitting can be avoided.

Table 7.4 shows the posterior statistical moments including mean values, standard deviations (STD) and modes. The mode of posterior samples for a certain calibration parameters is the value that appears most often, which is essentially the Maximum A Posteriori (MAP) estimation.

Figure 7.13 and 7.14 show the posterior pair-wise joint densities and marginal densities for the five physical model parameters, with and without considering the model discrepancy respectively. The marginal PDFs are evaluated using Kernel Density Estimation (KDE). The x

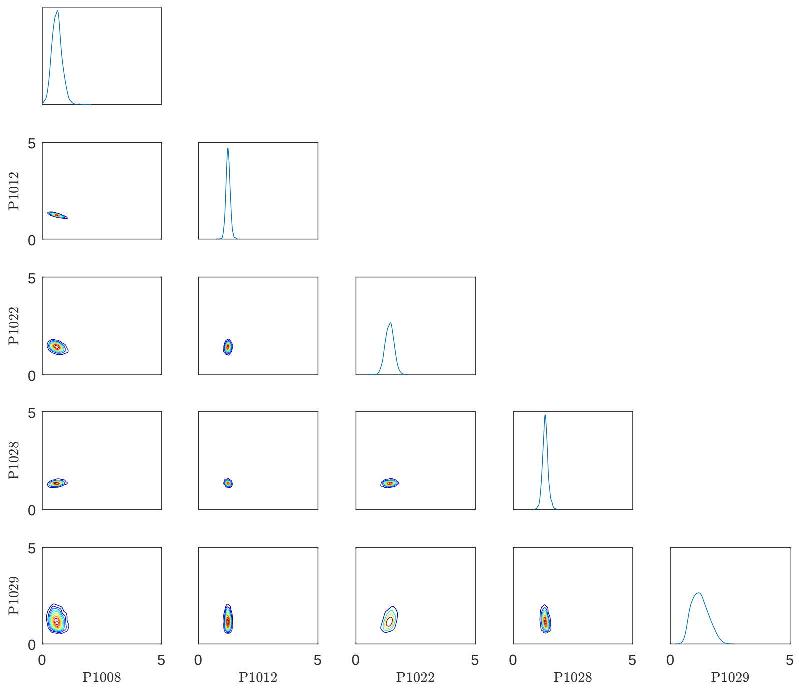


Fig. 7.13 Posterior pair-wise joint and marginal densities when model discrepancy is considered

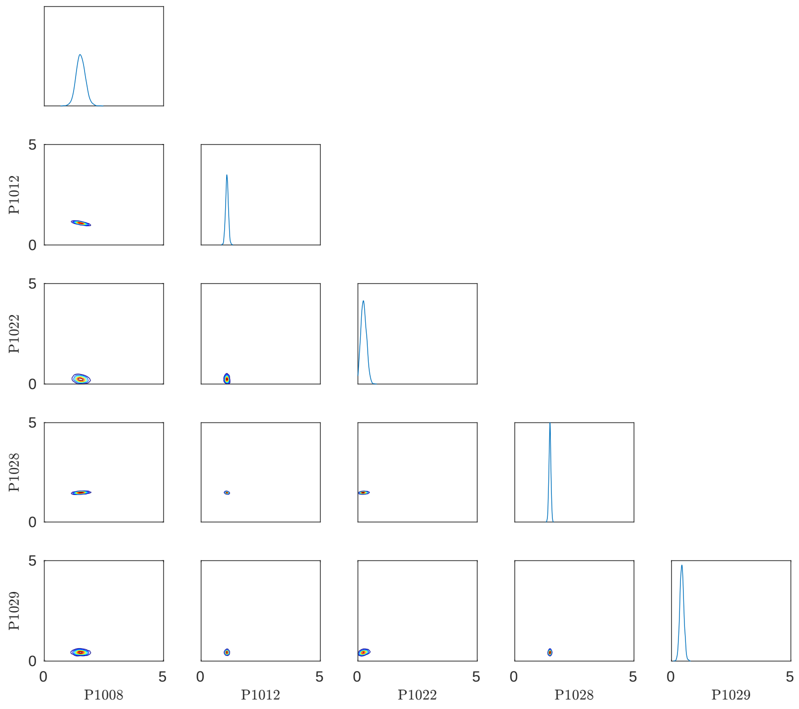


Fig. 7.14 Posterior pair-wise joint and marginal densities when model discrepancy is not considered

and y axes of joint densities and x axis of the marginal densities are the prior ranges. Apparently, the posterior distributions demonstrate a remarkable reduction in prior input uncertainties. These density plots are also useful for identifying potential correlations between the parameters, as well as the type of marginal distribution for each parameter. For example, highly linear negative correlation is observed between `P1008` and `P1012`. This indicates that in future forward uncertainty propagation studies, these input parameters should be sampled jointly, not independently, so that their correlation is captured.

By comparing results with and without model discrepancy in Table 7.4, Figure 7.13 and 7.14, it can be noticed that when model discrepancy is not considered, the posterior standard deviations are very small and pair-wise joint distributions are more concentrated. This fact is preferable in the sense that more uncertainty reduction is achieved. However, it is also an indication of potential over-fitting. At this point, we do not know which one of the results in Figure 7.13 and 7.14 is “closer” to the truth. Validation of TRACE based on these posteriors is needed to make a decision.

7.6 Results for Validation Using Posteriors

In this section, preliminary validation is performed (blocks connected by light blue arrows in Figure 2.6) for the posteriors (with and without mode discrepancy) achieved in Section 7.5.

7.6.1 Validate posteriors from inverse UQ

Figure 7.15 shows the following:

- Green symbol: $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}}) - \mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{0})$, TRACE prediction error when evaluated at the prior nominals.
- Blue symbol: $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}}) - \overline{\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})}$, difference between measurement data and mean values of TRACE evaluated at posterior samples. Model discrepancy (called “bias” in the figure) is not considered and the standard deviations of $\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})$ are plotted as errorbar;
- Red symbol: $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}}) - \overline{\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})}$, different between measurement data and mean values of TRACE evaluated at posterior samples. Model discrepancy is considered and the standard deviations of $\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})$ are plotted as errorbar;

Note that in Section 7.5, after burn-in and thinning there are 4,000 samples left. For validation if we run TRACE code at these samples the computational cost is still considerable. To avoid this, we can create another `GPcode` for TRACE with $(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})$ and

$\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})$ and training input and outputs. Again for each test in $\mathbf{x}^{\mathrm{VAL}}$, a maximin LHS design is generated with N_{design} samples and all together $N_{\mathrm{VAL}} \times N_{\mathrm{design}}$ new TRACE runs are needed to train the GP emulator. Because N_{VAL} is two times larger than N_{IUQ}, one may question that the computational cost will be high. The simulation cost is not really high due to two reasons: (1) the fact that $N_{\mathrm{VAL}} \approx 3 \times N_{\mathrm{IUQ}}$ means that a smaller value of N_{design} (e.g. 10) will be sufficient to build an accurate GP emulator for TRACE; (2) The $\boldsymbol{\theta}^{\mathrm{Posterior}}$ has significant uncertainty reduction compared with $\boldsymbol{\theta}^{\mathrm{Prior}}$ so that much fewer design samples are needed.

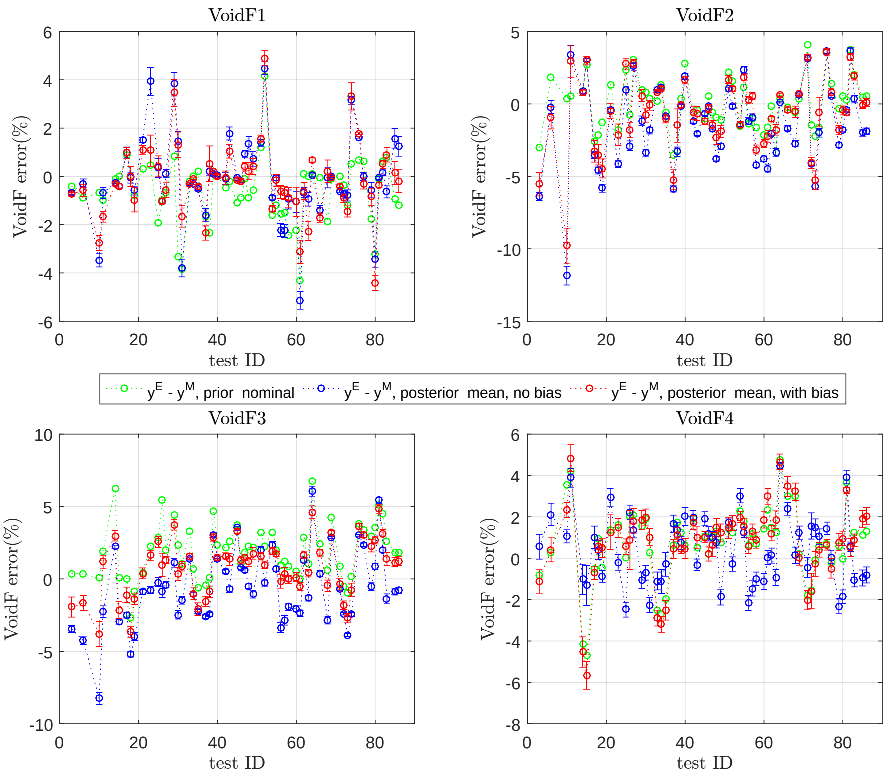


Fig. 7.15 Comparison of validation data with TRACE simulation based on prior and posterior

It can be observed from Figure 7.15 that when model discrepancy is considered, the mean of TRACE evaluated at posterior samples $\overline{\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})}$ are generally closer to $\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{0})$. Recall that $\boldsymbol{\theta}^{0}$ represents our current best knowledge and recommended by TRACE model developers, any audacious change of $\boldsymbol{\theta}^{0}$ may be questionable. For example, without model discrepancy, the inverse UQ process suggests changing `P1022` to 0.2452 and `P1029` to 0.4321 which may be unacceptable by model developers and other users. Although with model discrepancy, inverse UQ still suggests $23\% - 41\%$ change from $\boldsymbol{\theta}^{0}$ to $\boldsymbol{\theta}^{\mathrm{Posterior}}$ mean, the standard deviations of $\boldsymbol{\theta}^{\mathrm{Posterior}}$ are much larger.

Also, it is generally true (with a few exceptions) that $\mathbf{y}^{\mathrm{E}}(\mathbf{x}^{\mathrm{VAL}}) - \overline{\mathbf{y}^{\mathrm{M}}(\mathbf{x}^{\mathrm{VAL}}, \boldsymbol{\theta}^{\mathrm{Posterior}})}$ are larger when model discrepancy is not considered. This is because without model discrepancy, all the connections from green arrows in Figure 2.6 are gone, the posteriors $\boldsymbol{\theta}^{\text{Posterior}}$ achieved using $\mathbf{y}^{\text{E}}\left(\mathbf{x}^{\text{IUQ}}\right)$ are completely ignorant of $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}})$. The posteriors are slightly over-fitted to $\mathbf{y}^{\text{E}}\left(\mathbf{x}^{\text{IUQ}}\right)$ such that when TRACE is evaluated at $\boldsymbol{\theta}^{\text{Posterior}}$, larger $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^{\text{Posterior}})$ are observed.

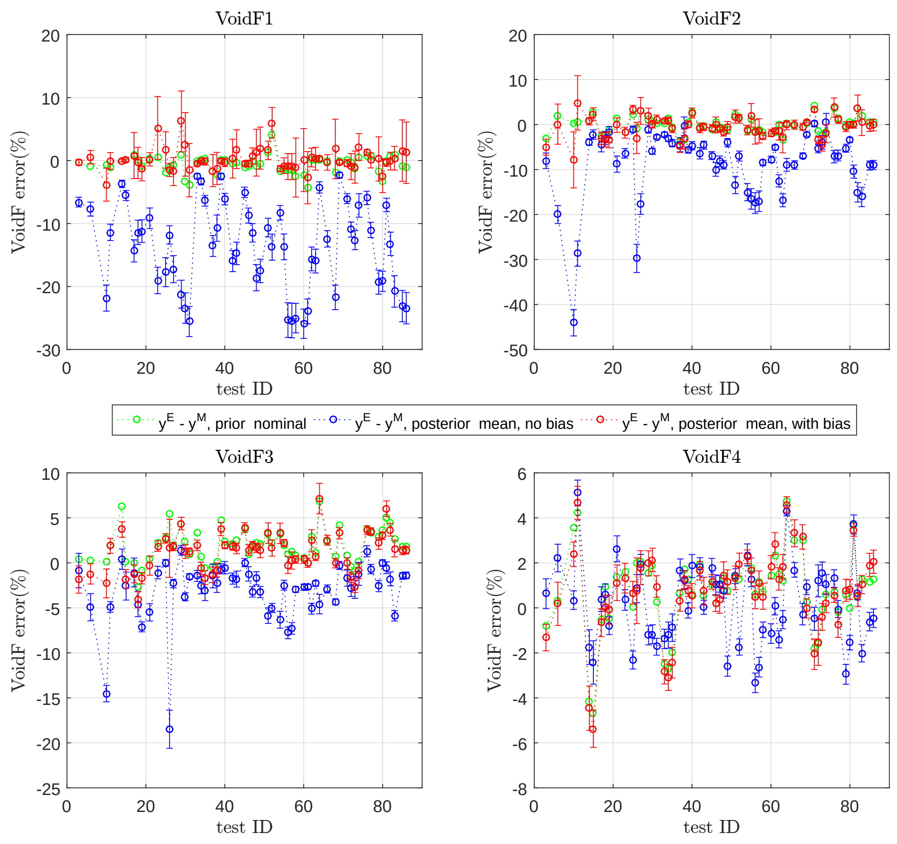


Fig. 7.16 Comparison of validation data with TRACE simulation based on prior and posterior, when inverse UQ is only performed with VoidF3 and VoidF4.

However, the over-fitting is not obvious. To demonstrate a scenario with strong over-calibration, we repeated the whole inverse UQ process using only two QoIs: VoidF3 and VoidF4. These two QoIs are picked because `P1008` and `P1012` have trivial influence on them, which means these two parameters will not be identifiable with only VoidF3 and VoidF4. Figure 7.16 shows the same comparison with Figure 7.15 when only VoidF3 and VoidF4 are used for inverse UQ. Tremendous TRACE prediction errors $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}}) - \mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^{\text{Posterior}})$ are observed when model discrepancy is not considered, especially for VoidF1 and VoidF2. The posteriors are so over-fitted to $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{IUQ}})$ that they lead to bad agreement with $\mathbf{y}^{\text{E}}(\mathbf{x}^{\text{VAL}})$. However, our proposed treatment of model discrepancy demonstrates that it can avoid over-fitting in this scenario. The only influence on $\mathbf{y}^{\text{M}}(\mathbf{x}^{\text{VAL}}, \boldsymbol{\theta}^{\text{Posterior}})$ is that the standard deviations have increased.

7.6.2 Fitted posterior distributions

After demonstrating the validity of the improved modular Bayesian approach. The posterior $\boldsymbol{\theta}^{\text{Posterior}}$ when model discrepancy is considered is the preferred results for inverse UQ. To make these results more applicable to future forward UQ, sensitivity analysis and validation studies, the posterior samples for each physical model parameter can be fitted to certain well-known distributions, e.g. Gaussian distribution. Figure 7.17 and Table 7.5 show the fitted distributions for each physical model parameter and the parameters associated with each distribution, i.e. mean (μ) and standard deviation (σ) for normal distribution, shape (α) and scale (β) parameter for Gamma distribution, non-centrality (s) and scale (σ) parameter for Rician distribution. All the fitted distributions are accepted by Kolmogorov–Smirnov test at the 5% significance level. Figure 7.18 shows that good agreement can be achieved between the empirical cumulative distribution function (CDF) and fitted distribution CDF for every physical model parameter. Finally, Table 7.6 reports the correlation coefficients between the five calibration parameters.

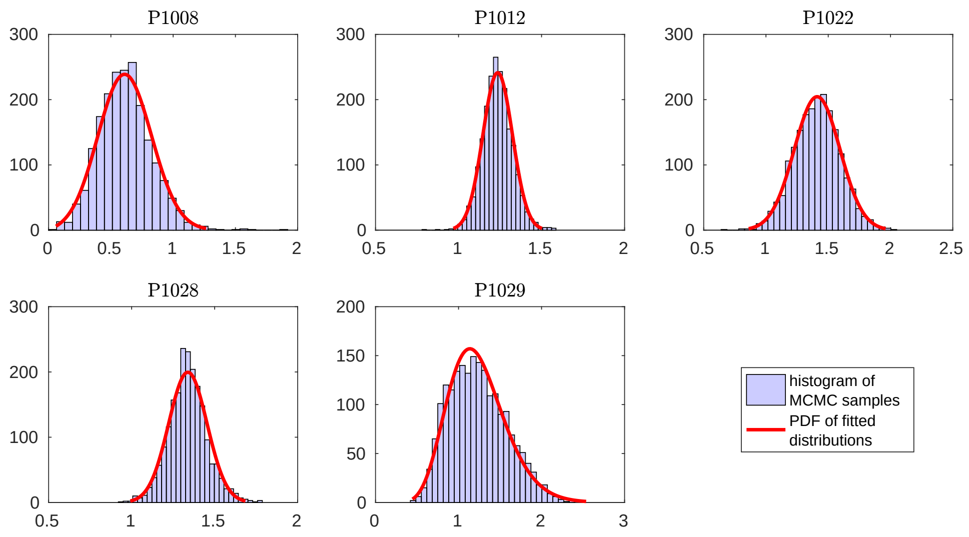


Fig. 7.17 Fitted posterior marginal probability densities

Table 7.5 Fitted distribution type and distribution parameters

Parameter	Distribution type	Distribution parameter 1	Distribution parameter 2
P1008	Rician	$s = 0.5709$	$\sigma = 0.2218$
P1012	Gaussian	$\mu = 1.2358$	$\sigma = 0.0890$
P1022	Gaussian	$\mu = 1.4110$	$\sigma = 0.1833$
P1028	Gaussian	$\mu = 1.3385$	$\sigma = 0.1155$
P1029	Gamma	$\alpha = 12.6511$	$\beta = 0.0975$

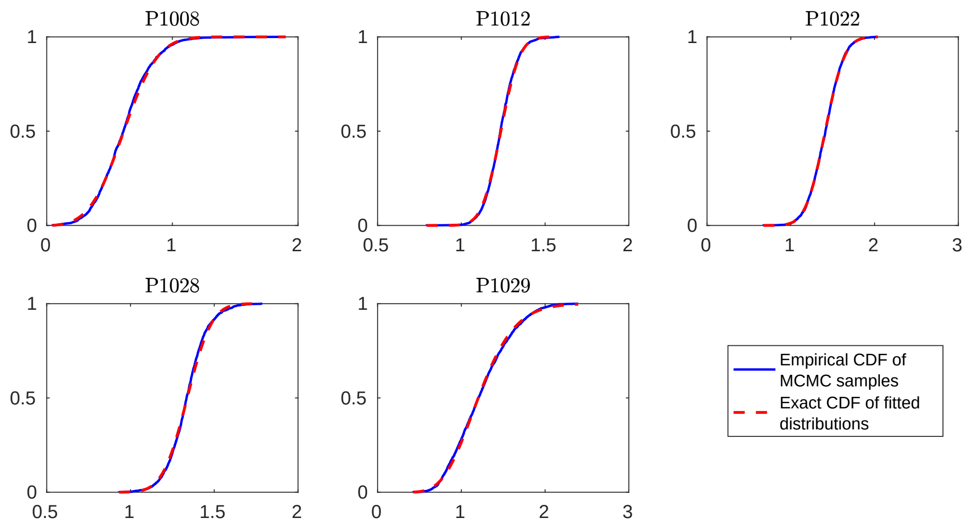


Fig. 7.18 Comparison of empirical CDFs and fitted distribution CDFs.

Table 7.6 Correlation coefficients of five physical model parameters

Parameter	P1008	P1012	P1022	P1028	P1029
P1008	1.0000				
P1012	-0.8338	1.0000			
P1022	-0.3543	0.0969	1.0000		
P1028	0.3264	-0.2121	0.1978	1.0000	
P1029	-0.1941	0.0251	0.4047	-0.2246	1.0000

7.7 Discussions

The improved modular Bayesian approach presented in Section 2.3 has been successfully demonstrated to be able to inversely quantify the uncertainties in calibration parameters and avoid over-fitting in the meantime. The proposed inverse UQ approach is robust in the sense that satisfactory results can be achieved even only partial QoIs data are used. GP-based metamodel can greatly reduce the computational cost by several orders of magnitude, as shown in Table 7.7.

Table 7.7 Simulation cost for each step in the workflow shown in Figure 2.6

Simulation cost	TRACE full model	GPbias	GPcode for inverse UQ	GPcode for validation
Cost in TRACE runs	50,000	86	380	590
Extra CPU time		36.7 mins		

However, there are some limitations in this study that need further investigations:

1. Inverse UQ only takes 5 of the 36 TRACE physical model parameters for this study, because only these 5 parameters are significant for BFBT benchmark problem. To inversely quantify the uncertainties in other parameters, different benchmark data are required. For example, parameters `P1034` and `P1035` will need experimental data that involves reflooding. However, by separated inverse UQ for different groups of parameters, the correlations between the groups will not be quantified. For instance, if parameter groups A and B are significant to benchmarks A and B respectively, then inverse UQ will only result in correlations of parameters within each group but not across the two groups.

2. The current problem has four-dimensional QoIs (VoidF1 – VoidF4) which are supposed to be correlated. However, because we do not have enough information to quantify the correlations between them, they are assumed to be independent to each other when constructing the multi-dimensional GP emulator `GPbias` and `GPcode`. As a result, all the three components of the covariance matrix of the likelihood function $\mathbf{\Sigma}_{\text{exp}} + \mathbf{\Sigma}_{\text{bias}} + \mathbf{\Sigma}_{\text{code}}$ are diagonal matrices. The inverse UQ is more solid if the correlation structure of the QoIs can be quantified and incorporated in the evaluation of the likelihood.

3. In this study, we used $\alpha = 25\%$ of the experimental data for inverse UQ and the rest for validation. Even though the sequential algorithm for TSA is rigorous and justified, further numerical investigation is needed to see if different values can cause significant changes in the inverse UQ results.

4. In Section 7.6, "graphical validation" method is used which is only qualitative. It provides very limited quantitative information of the variation of the code performance with the design variables. We are working on quantitative validation metrics that can account for the correlations between different QoIs.

5. Model updating is an iterative process. Sequential inverse UQ of the computer model can be done by using the current posterior as new prior for the next inverse UQ process.

6. Finally, there is an important issue for inverse UQ that is not addressed in this work: the "identifiability" problem. In Section 2.3 we have briefly commented on this issue. Inverse UQ problems are usually ill-posed. Many different combinations of the model discrepancy and parameter variability can account for the same amount of error between code simulation and experimental observation, making the true values of the calibration parameters not "identifiable". We have discovered some connections between the sensitivity and identifiability, which will be published in the near future once more extensive numerical tests are finished.

Chapter 8

APPLICATION TO BISON FISSION GAS RELEASE MODEL

In this chapter, we quantified the uncertainties for the input parameters of fuel performance code BISON fission gas release (FGR) model, based on Risø-AN3 benchmark time series FGR measurement data. GP metamodel is used during MCMC sampling. This specific problem has a few features that pose challenges for inverse UQ, i.e., (1) time series data correspond to high-dimensional correlated outputs; (2) we have only one measurement setting available to use which is insufficient to provide a model for model discrepancy of BISON; (3) the time series from BISON and measurement data have drastic difference, which can cause unexpected random walks if the FGR data is used directly. The work presented in this chapter is also published in [153] [155].

8.1 Problem Definition

Nuclear reactor fuel performance studies the thermo-mechanical behavior of fuel rods and verify their compliance with safety criteria under both normal operation and accidental conditions. Various complex phenomena need to be considered in nuclear reactor fuel performance analysis [147], for example: (1) for fuel: fission product swelling, densification, thermal and irradiation creep, fracture, and fission gas production and release; (2) for cladding: cladding plasticity, irradiation growth, and thermal and irradiation creep; (3) for others: gap heat transfer, mechanical contact, and evolution of the gap/plenum pressure with plenum volume, gas temperature, and fission gas addition, etc. Example of some popular fuel performance codes are BISON [147], TRANSURANUS [76], ENIGMA [72], FRAPCON [113] and FALCON [43].

8.1.1 Fission Gas Behavior in Nuclear Fuel

In this chapter, we focus on the behavior of the fission gases xenon and krypton in uranium dioxide fuel. The fundamental physical processes, which control the kinetics of FGR in irradiated UO_2, may be outlined as follows [105] [106]:

1. Fission gas atoms generated in the fuel grains diffuse towards the grain boundaries through repeated trapping in and irradiation-induced resolution from nanometer-size intra-granular bubbles.
2. Gas accumulates at grain faces as a result of intra-granular diffusion and of gas sweeping by moving grain boundaries as the grain growth process takes place.
3. Although a part of the gas atoms at the grain faces is dissolved back to the grain interior by irradiation, the majority diffuses into grain-face bubbles acting to increase the bubble internal pressure and generally maintaining bubbles in a non-equilibrium state.
4. Micron-size grain-face bubbles grow with inflow of gas atoms and with absorption of vacancies driven by the bubble over-pressure.
5. Bubble growth brings about bubble coalescence and inter-connection, eventually leading to the formation of a tunnel network through which a fraction of the gas is released to the fuel rod free volume (thermal release).

The complex behavior of the fission gases xenon and krypton in UO_2 significantly affects the thermo-mechanical performance of the nuclear fuel rods employed in current LWRs due to the following reasons [105] [106] [107]:

1. The fission gases tend to precipitate into bubbles after production which results in fuel swelling and promotes fuel rod gap closure and the ensuing Pellet-Cladding Mechanical Interaction (PCMI).
2. The released fission gas accumulates in fuel rod free volume (fuel-cladding gap), causing pressure build-up and thermal conductivity degradation of the fuel rod filling gas (helium).
3. The precipitated gas bubbles in fuel rod also has a negative effect on the fuel thermal conductivity, and consequently the temperature distribution in the fuel pellet.
4. The increase of fuel temperature will in turn lead to higher amount of released fission gases which forms a positive feedback. Eventually the rod will fail due to cladding ballooning and cladding burst.

The accurate modeling of fission gas behavior in nuclear fuel performance simulation is vital considering its detrimental nature. However, the numerical analysis of FGR and swelling involves treatment of several complicated and interrelated physical processes, which inevitably depend on uncertain input parameters. For example, the state-of-the-art fuel performance code BISON [147] incorporates an advanced physics-based FGR model that depends on several model parameters whose uncertainties are only known by expert judgement [107]. This poses difficulties in the uncertainty and sensitivity analysis of BISON for other applications. The objective of this chapter is to inversely quantify the uncertainties associated with such model parameters in BISON FGR model based on available experimental data.

8.1.2 BISON FGR Model

BISON is a finite element-based nuclear fuel performance code developed at Idaho National Laboratory (INL) [147]. BISON is built on MOOSE (Multi-physics Object Oriented Simulation Environment) framework [41], which is a parallel computational framework designed for rapid production of new simulation tools. BISON solves the fully-coupled equations of thermo-mechanics and species diffusion in an implicit and parallel way, for either 1-D spherical, 2-D axisymmetric or 3-D geometries. It incorporates a wide variety of material models for both fuel and zircaloy cladding. Other capabilities of BISON include applicability to both steady-state and transient conditions.3 Extensive Verification, Validation and assessment work has been done with BISON code, see [56] [146] [108].

Given the unfavorable nature of fission gases as a potential life-limiting factor of nuclear fuel rods, a reliable modeling of FGR and swelling represents a primary requisite for fuel performance codes. Previously, the modeling of FGR in nuclear fuel performance codes have been relying on empirical models which are not applicable beyond their range of calibration [105]. Recently a more efficient and flexible physics-based FGR and swelling model that can describe a wider range of reactor operation conditions have been implemented in BISON [106]. This new model is called “Simple Integrated Fission Gas Release and Swelling (SIFGRS)”, which uses a physics-based modeling approach described in [105] [106]. Several fundamental features of fission gas behavior are incorporated in this model, including gas generation, diffusion and precipitation in grains, growth and coalescence of gas bubbles at grain faces, grain growth and grain boundary sweeping, thermal, athermal, and transient gas release. Furthermore, in contrast with historical empirical approaches, the SIFGRS model allows the simulation of the FGR and swelling as inherently coupled processes, while remaining a level of complexity suitable for application to engineering scale fuel performance analysis [105] [106].

The BISON SIFGRS model includes five uncertain input parameters, all of which are scale parameters as shown in Table 8.1. The lower and upper bounds are suggested by SIFGRS developers in a recent BISON uncertainty and sensitivity study [107]. Such uncertainty bounds specification is based on the scatter in the available experimental data from a fairly

Table 8.1 Uncertain input parameters in BISON SIFGRS model

Parameter (scale factor)	Description	Lower bound	Upper bound	Nominal
Temperature	`temperature_scalef`	0.95	1.05	1.0
Grain radius	`grainradius_scalef`	0.4	1.6	1.0
Intra-granular gas atom diffusion coeff.	`igdiffcoeff_scalef`	0.1	10.0	1.0
Intra-granular resolution parameter	`resolutionp_scalef`	0.1	10.0	1.0
Grain-boundary diffusion coeff.	`gbdiffcoeff_scalef`	0.1	10.0	1.0

extensive literature review and is consistent with the information in the open literature. Normal distributions are assumed for the first two parameters. The last three parameters adopt log-uniform distributions so that the logarithms of them follow uniform distribution on $[-1,1]$ [107]. We can see that the uncertainty ranges for the last three parameters have lower and upper bounds that differ by a factor of 100. Apparently a more robust and appropriate specifications of the input uncertainties should be sought that are consistent with available experimental data. This is the primary mission of the inverse UQ analysis in this study, and we have chosen Risø-AN3 benchmark which includes on-line, time-dependent measurement of FGR.

8.2 Risø-AN3 Experimental Data

8.2.1 Risø-AN3 Benchmark

The Risø-AN3 experiment is one of the fuel rod irradiation experiments from the International Fuel Performance Experiments (IFPE) database [73] [122]. It comprises base irradiation in the BIBLIS A PWR (Germany) and ramp test in the DR3 research reactor at Risø (Denmark). The mother rod, CB8, was irradiated over four reactor cycles up to a burnup of about 40.9 GWd/t, and re-fabricated to a shorter length. The re-fabricated rod, CB8-2R, was instrumented with a fuel centerline thermocouple and a pressure transducer. Table 8.2 presents the specifications for the re-fabricated rod which was extracted from an irradiated full length rod. The thermocouple hole was at the top of the fuel rod and did not penetrate the entire fuel stack.

The power history for the base irradiation of the full length fuel rod is shown in Figure 8.1. Figure 8.2 presents the ramp test power history of the re-fabricated rod. The ramp test takes about 72 hours with a peak power of approximately 40 kW/m, resulting in a final burnup of 41.8 GWd/t. A prescribed axial power profile for both base irradiation and ramp test was provided in the FUMEX-II data [73], along with the measured clad surface temperature as a function of time which was used as a boundary condition for this simulation. Additional reactor operation conditions can be found in BISON assessment manual section M [108].

Table 8.2 Specifications of Risø-AN3 re-fabricated test rod (TD means Theoretical Density)

Description	Property	Value	Unit
Material	Fill gas	He	
	Mother rod fill gas pressure	2.31	MPa
	Re-fabricated rod fill gas pressure	1.57	MPa
	Fuel material	UO_2	
	Fuel enrichment	2.95	wt% 235U
	Fuel density	93.74	%TD
	Cladding material	Zircaloy-4	
	Average grain radius	4.7	µm
Geometry	Overall length	39.058	cm
	Fuel stack height	286.8	mm
	Nominal plenum height	60.96	mm
	Pellet inner diameter	2.5	mm
	Pellet outer diameter	9.053	mm
	Dish diameter	6.65	mm
	Dish depth	0.13	mm
	Chamfer width	0.46	mm
	Chamfer depth	0.16	mm
	Cladding inner diameter	9.258	mm
	Cladding outer diameter	10.81	mm

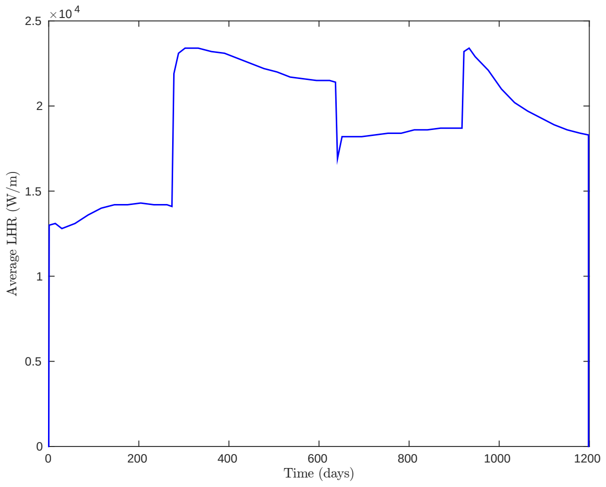


Fig. 8.1 Risø-AN3 benchmark base irradiation power history.

8.2.2 BISON Modeling of Risø-AN3 Benchmark

The BISON code incorporating the most recent SIFGRS model was applied to the analysis of Risø-AN3 irradiation experiment, using nominal values for the five uncertain scale parameters.

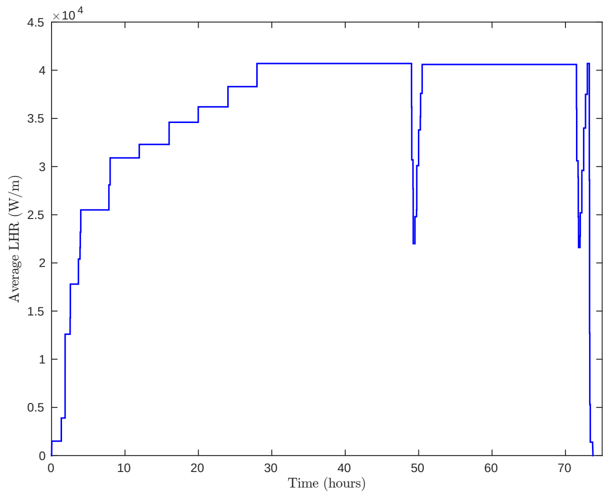


Fig. 8.2 Risø-AN3 benchmark ramp test power history.

We used a 2D axisymmetric model for the fuel. Other main models and characteristics of the BISON code can be found in [147]. We followed the implementation details described in BISON assessment manual [108], for example, the geometry and mesh, material and behavioral models, boundary and operating conditions. The simulations were performed using INL's high performance computing cluster. For each simulation, it takes around 1.75 hours using 32 processors.

Figure 8.3 shows the comparison of fuel temperature at thermocouple position during ramp test from Risø-AN3 measurement data and BISON prediction. The fuel centerline temperature is taken at a node approximately 36.4 mm from the top of the fuel stack. The comparison indicates that BISON code is able to capture the fuel temperature evolution within 10% of the measurement throughout the duration of the transient. However, BISON systematically under-predicts the fuel centerline temperature.

Figure 8.4 presents the comparison of the rod internal pressure. BISON over-predicts the rod internal pressure by 20% - 40% over the ramp test. Rod internal pressure is affected by many factors such as plenum pressure and volume, rod axial growth, fuel and cladding creep, etc. It is unsure which one is the main factor that causes the over-prediction. The burst of rod internal pressure around 50 hours after the start of transient may be partially associated with gap and cracking opening during the sudden power drop.

Figure 8.5 shows the comparison of measured and BISON simulated FGR during the ramp test. The FGR is defined as the ratio between the amount of fission gas released and generated in the irradiated fuel rod. BISON under-predicts the FGR over the most of the ramp test. It is able to qualitatively reproduce the rapid increase of FGR around 50 hours after the start of transient, albeit only by a much lower magnitude. A possible explanation, as pointed out in [146], is that

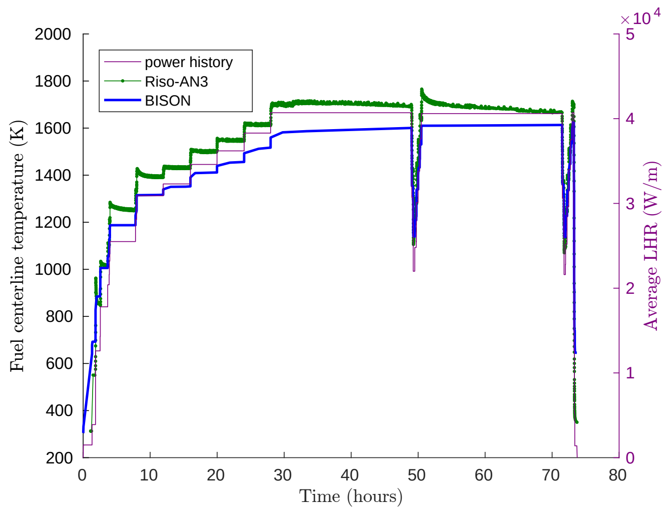


Fig. 8.3 Comparison of fuel centerline temperature from Risø-AN3 measurement and BISON simulation with nominal values of uncertain input parameters.

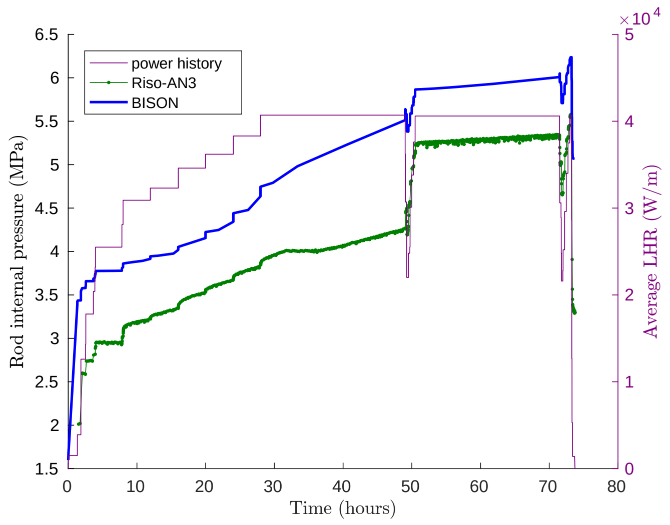


Fig. 8.4 Comparison of rod internal pressure from Risø-AN3 measurement and BISON simulation with nominal values of uncertain input parameters.

the magnitude of the release burst measured by pressure transducer is over-estimated. Another reason is that the gap and cracking opening during the sudden power drop can cause delayed detection of gas released from the fuel prior to the transient [73] [105].

Considering the difficulty and inherent uncertainties of simulating FGR during ramp tests by means of engineering-scale fuel performance codes, it is commonly regarded as satisfactory when the code FGR predictions deviate less than a factor of 2 from the measured values [107] [146]. Actually the accuracy of the FGR calculation for Risø-AN3 obtained by BISON as shown in Figure 8.5 is similar or better than the state-of-the-art [73] [146].

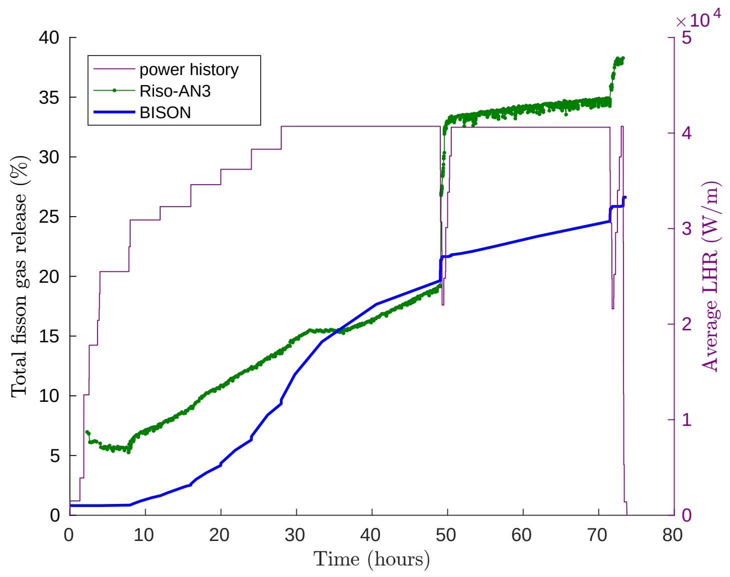


Fig. 8.5 Comparison of FGR time series from Risø-AN3 measurement and BISON simulation with nominal values of uncertain input parameters.

Even though fuel centerline temperature and rod internal pressure measurement data are also available in Risø-AN3 benchmark, they present immediate responses to the transient power changes and thus have multiple discontinuities when the power suddenly changes, as shown in Figure 8.3 and Figure 8.4. FGR evolution with time is relatively smoother and it is the primary quantity that we are interested in. Therefore we choose the time-dependent FGR measurement data for the inverse UQ study.

8.3 Dimension Reduction

There are two unresolved issues before using the Risø-AN3 FGR time series data for inverse UQ of BISON FGR model:

1. The experimental FGR data is time-dependent, which corresponds to a multivariate output. As these output variables are highly correlated, a multivariate GP emulator is required instead of independent emulators for each variable. Making separate GP model for each variable is too cumbersome and it will need special treatment to account for the high correlation.

2. It can be seen in Figure 8.5 that the shapes of FGR evolution from BISON and Risø-AN3 have drastic difference, mainly due to the under-estimation of the burst release of FGR when power drops suddenly. Moreover, at a different input combinations of the five scale parameters, BISON FGR simulations will have similar shapes. Therefore, inverse UQ is

not expected to find an optimal “posterior” PDF for the random input parameters such that the BISON calculations will “match” the measurement. Clearly, this drastic difference is caused by model discrepancy $\delta(\mathbf{x})$. However, we do not have enough available field data to train a GP model for $\delta(\mathbf{x})$ In this case, the magnitude of the measurement error is insufficient to account for the misprediction, and we are likely to have a poor model fit during the inverse UQ process.

The first problem has been tackled with dimension reduction by Principal Component Analysis (PCA) in previous research [61] [62] [144] [166] [142]. PCA is the projection of high-dimensional data onto a lower-dimensional subspace (also called principal subspace) such that the projected data has maximized variance and are uncorrelated [124]. PCA can be done through eigen-decomposition of the covariance matrix of the data. However, Singular Value Decomposition (SVD) of the data matrix is often preferred for numerical reasons.

8.3.1 Principal Component Analysis

Suppose we have picked p points from BISON simulated FGR evolution with time. The FGR values at those points will correspond to p output variables which is usually a large number resulting in a high-dimensional problem (e.g., 100). We believe that it not necessary to represent the FGR time series with so many variables. To learn the underlying evolution pattern of the time series, we sample the uncertain input parameters N times, run BISON at these inputs and collect the FGR data which form a $p \times N$ data matrix $\mathbf{A}$. We are aware of the high correlation between the rows of this data matrix and we would like to de-correlate the data with a linear transformation $\mathbf{PA} = \mathbf{B}$, where $\mathbf{P}$ is a $p \times p$ transformation matrix, $\mathbf{B}$ is a $p \times N$ data matrix and the rows of $\mathbf{B}$ represent uncorrelated new variables. PCA is the process to find $\mathbf{P}$ and the steps are shown below:

1. Center the original data matrix $\mathbf{A}$. Define $\mathbf{u}$ as the column vector of the row means. We get the centered data matrix $\mathbf{A}_{\text{centered}}$ by subtracting $\mathbf{u}$ from each column of $\mathbf{A}$.

2. Find the SVD of $\mathbf{A}_{\text{centered}}$:

$$\mathbf{A}_{\text{centered}} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{\top} \tag{8.1}$$

   where

   - $\mathbf{U}$ is a $p \times p$ orthogonal matrix whose columns are the left-singular vectors of $\mathbf{A}_{\text{centered}}$
   - $\mathbf{V}$ is a $N \times N$ orthogonal matrix whose columns are the right-singular vectors of $\mathbf{A}_{\text{centered}}$

- $\mathbf{\Lambda}$ is a $p \times N$ diagonal matrix whose diagonal entries are the singular values of $\mathbf{A}_{\text{centered}}$
- The non-zero singular values are the square roots of the non-zero eigenvalues of $\mathbf{A}_{\text{centered}}\mathbf{A}_{\text{centered}}^\top$ or $\mathbf{A}_{\text{centered}}^\top\mathbf{A}_{\text{centered}}$.
- The magnitude of the diagonal entries of $\mathbf{\Lambda}$ are arranged in descending order and large singular values points to important features in data matrix $\mathbf{A}_{\text{centered}}$ [124].

3. Choose $\mathbf{P} = \mathbf{U}^\top$, then we have:

$$\mathbf{P}\mathbf{A}_{\text{centered}} = \mathbf{U}^\top\mathbf{A}_{\text{centered}} = \mathbf{\Lambda}\mathbf{V}^\top = \mathbf{B} \tag{8.2}$$

   Apparently the covariance matrix of $\mathbf{B}$ is diagonal. The rows of $\mathbf{P}$ (which are the columns of $\mathbf{U}$) are called the **Principal Components (PCs)**, also known as **loadings**. The transformed variables (rows of $\mathbf{B}$) are called **PC scores**. PC scores are the representations of the original data $\mathbf{A}_{\text{centered}}$ in the principal subspace. The columns of $\mathbf{B}$ correspond to observations. The eigenvalues of the covariance matrix of $\mathbf{A}_{\text{centered}}$ are called **PC variances**, which correspond to the square of the diagonal entries in $\mathbf{\Lambda}$.

4. Decide on the dimension of the principal subspace p^* which is much smaller than p. It is quite normal that the PC variances decrease very quickly. PCs with larger associated variances represent important structure, while those with lower variances represent noise [57]. A widely used criterion is that we only keep the first p^* PCs whose corresponding PC variances can account for over 95% or 99% of the total variance. The first p^* PCs form a $p^* \times p$ transformation matrix $\mathbf{P}^*$:

$$\mathbf{P}^*\mathbf{A}_{\text{centered}} = \mathbf{B}^* \tag{8.3}$$

   where $\mathbf{B}^*$ is a $p^* \times N$ matrix whose rows represented new variables after dimension reduction. Now we have reduced the number of variables from p to p^*.

Once we have the PCs as rows of matrix $\mathbf{P}^*$ and PC scores stored in matrix $\mathbf{B}^*$, we build the GP metamodel for the p^* PC scores instead of the p output variables in the original space (as shown by step 1 in Figure 8.6). The data matrix $\mathbf{B}^*$ will serve as the training sample for the GP metamodel. The GP prediction for a certain input will be a $p^* \times 1$ vector $\mathbf{b}^*$, where the "$*$" superscript is used to indicate that it is a prediction in PC subspace. We can transformed it back to the original space with $\mathbf{P}^*$:

$$\mathbf{a} = \mathbf{P}^{*\top}\mathbf{b}^* + \mathbf{u} \tag{8.4}$$

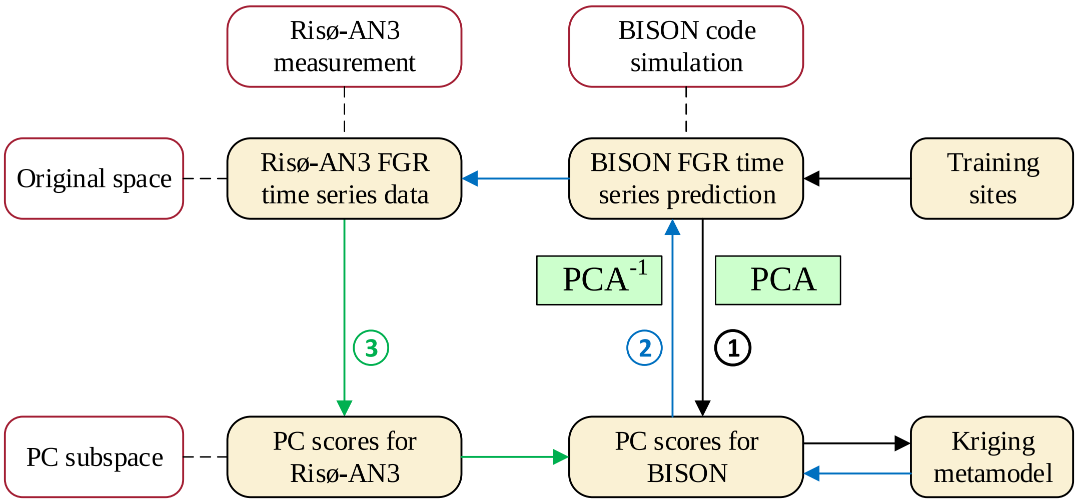


Fig. 8.6 Workflow and data processing steps.

where is a $p \times 1$ vector represents the FGR time series in the original space. Note that we need to add the mean vector $\mathbf{u}$ back. In all the previous work that used PCA dimension reduction for Bayesian calibration [61] [62] [144] [166] [142], the metamodel predictions for the PC scores were transformed back to the original space (as shown by step 2 in Figure 8.6). However, in this research we do not perform the "backward transformation". Instead we also transformed the measurement data to the PC subspace and use the transformed data during the inverse UQ process (as shown by step 3 in Figure 8.6). The motivation will be illustrated at the beginning of Section 8.5, in which we will show that by this approach we can deal with the second unresolved issue listed at the beginning of this section.

8.3.2 Results for Dimension Reduction

Before using the Risø-AN3 FGR time series data for the inverse UQ of BISON FGR model, data smoothing is performed because the measurement values are noisy as shown in Figure 8.5, especially at the last 20 hours of the ramp test. MATLAB "lowess" function with a spanning parameter of 0.05 is used for the data smoothing. Such spanning parameter is just enough to make the FGR monotonically increases with time over the most of the ramp test without smearing out too much of the original data. Figure 8.7 shows the smoothed FGR time series data. Data smoothing was also performed in a previous deterministic calibration performed for BISON FGR model [166].

We picked $p = 100$ points from the FGR evolution that are equally spaced over the ramp test to represent p outputs in the original space. Such a number is also chosen in a previous study by Yankov [166]. Next we need to find the dimension p^* for the PC subspace. As we are

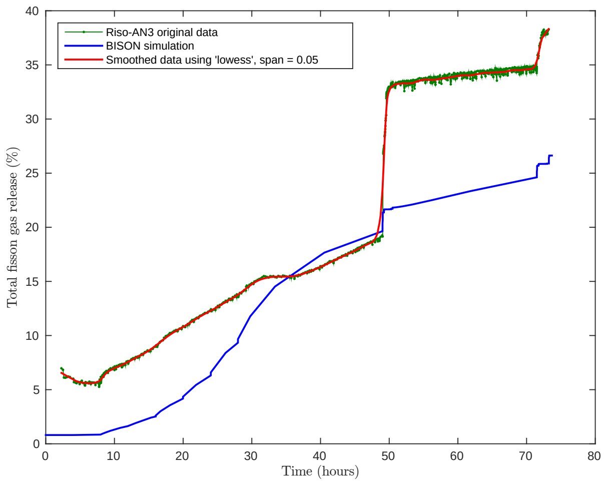


Fig. 8.7 Smoothing Risø-AN3 FGR time series data.

unaware of how many samples are needed for getting a good sense of the dimension reduction, we tried $N = 25$, 50, 100 and 200. Max-min LHS is used to generate samples over the input domain defined in Table 8.1. For each case of N we iterate the design 1000 times and select the one with the maximum minimum distance such that the input space is sufficiently explored.

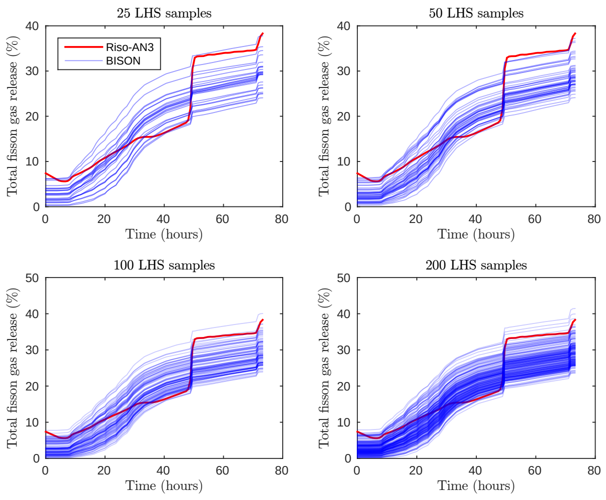


Fig. 8.8 Collection of BISON FGR simulation results from different max-min LHS designs.

The BISON FGR model developers choose uniform distributions for the first two parameters and log-uniform distributions for the last three parameters in Table 8.1 [107]. However, in that case equal number of samples will be generated for the ranges $[0.1, 1]$ and $[1, 10]$ for the last three parameters, resulting in insufficient exploration of the range $[1, 10]$. In the current study we choose uniform distributions for all the five input parameters during the LHS design. Figure 8.8 shows the collection of BISON FGR simulations with 25, 50, 100 and 200 LHS samples respectively. The figure confirms that by change the values of the five input parameters, the shapes of BISON FGR simulations remain similar. The FGR calculations for all the designs are close to “envelop” the Risø-AN3 FGR time series data, indicating that the choice of using uniform distributions over the prior ranges is reasonable.

Next we performed PCA for each of the four cases with different number of training samples. Figure 8.9 shows the decrease of the PC variances for the first few principal dimensions. It is apparent that the PC variances drop so quickly that the variances associated with higher index become trivial compared with the first few variances.

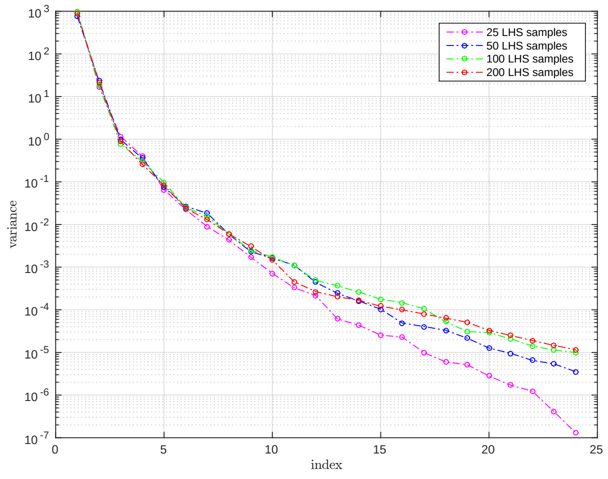


Fig. 8.9 Decrease of PC variances for the first few PC dimensions.

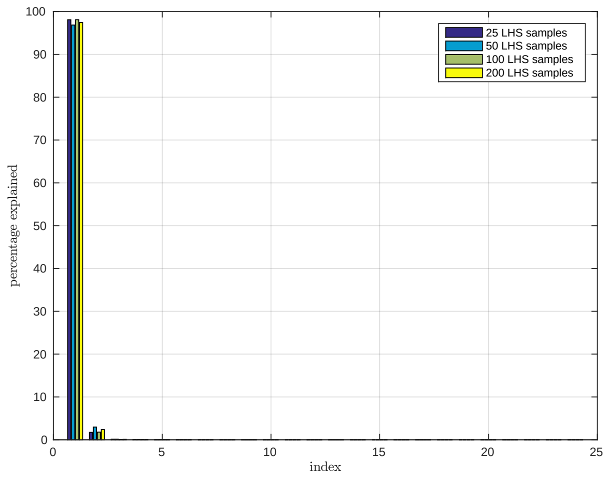


Fig. 8.10 Percentage of variation explained by first few PC dimensions.

Figure 8.10 shows the percentage of variation explained by the first few principal dimensions. The percentage of variation explained is the ratio of the PC variance in certain principal dimension over the sum of the variances in all the principal dimensions. Figure 8.11 shows the cumulative percentage of variation explained by the first few principal dimensions. The first principal dimension explains over 95% of all the variations. The first two PC dimensions together are able to account for over 99.7% percent of the total variation, suggesting that we only need to keep these two dimensions for the dimension reduction.

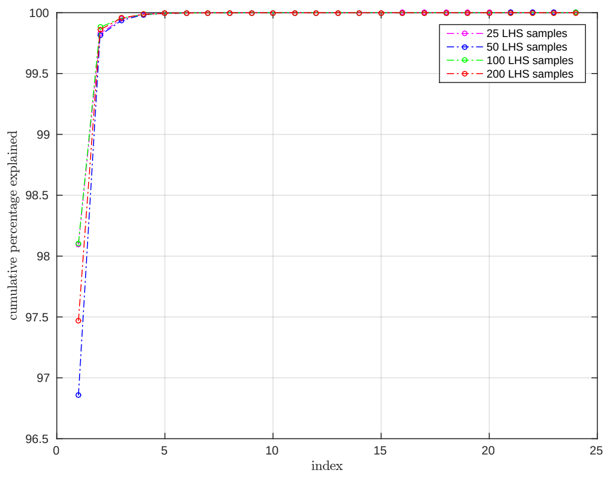


Fig. 8.11 Cumulative percentage of variation explained by first few PC dimensions.

In the following, we will build GP metamodels only for the first $p^* = 2$ PC scores, instead of build metamodels for $p = 100$ outputs in the original space. Since the dimension p^* of reduced space is larger than 1, we still have a multivariate problem. We can build a multivariate emulator, but building independent GP metamodels for each PC score often performs as well as a multivariate emulator because the PC scores are not correlated [144].

Figure 8.12 shows the first two PCs. Recall that each PC or loading vector is a $1 \times p$ vector, whose elements are essentially weighting factors for the linear transformation. The weighting factor in each PC has a one-to-one correspondence with the time that we have chosen for the p FGR outputs. There is no considerable difference between the PCs calculated using different sample sizes.

8.4 GP Metamodels for BISON

8.4.1 Building the GP Metamodels

In this subsection, we present the results for building and validating of GP metamodels. We used the PC scores from Section 8.3 as training samples. Again we considered different training sizes to study how many training samples are sufficient to build an accurate GP metamodel. In the current research we use the Matérn 5/2 correlation kernel, constant trend functions and MLE hyperparameter estimation method. Such choices are the most popular combinations in

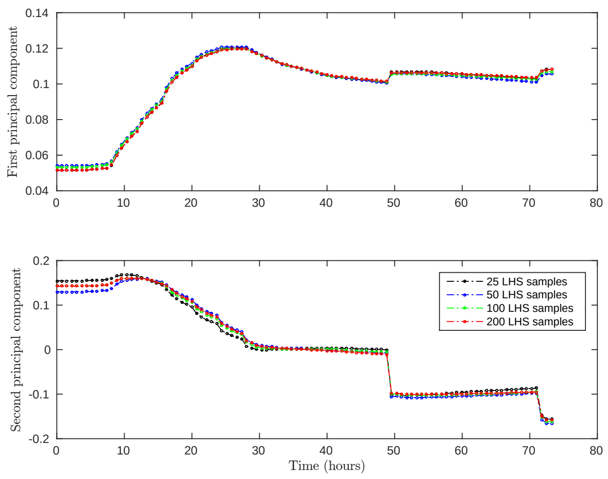


Fig. 8.12 First two PCs after dimension reduction.

literature. We did perform a comparative study for different correlations kernels and different trend functions and the current selection shows the best results, see Figure 8.13 - Figure 8.15.

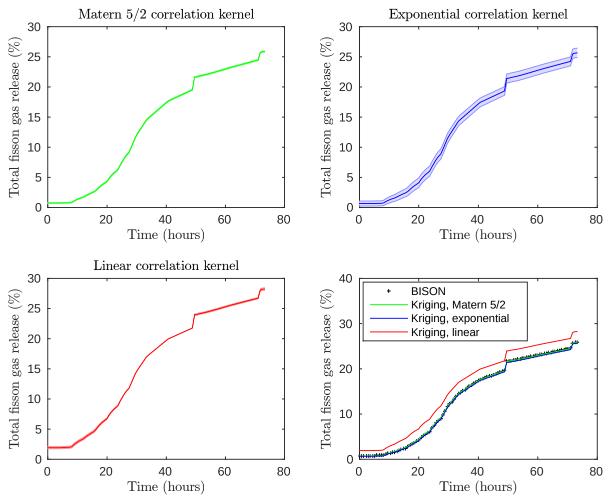


Fig. 8.13 Prediction at nominal values of inputs using GP metamodels built with different correlations kernels.

As a tentative study to evaluate the accuracy of GP metamodels with different correlations kernels and trend functions, we only look at the accuracy of the predictions at BISON nominal inputs.

1. Figure 8.13 compares the Matérn 5/2, exponential and linear correlation kernels, all with constant trend functions and 100 training samples. Apparently exponential correlation kernel results in larger MSE while linear correlation kernel leads to bad prediction.

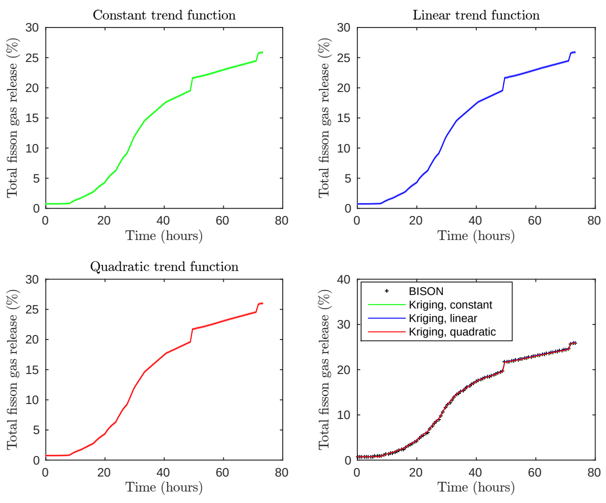


Fig. 8.14 Prediction at nominal values of inputs using GP metamodels built with different trend functions.

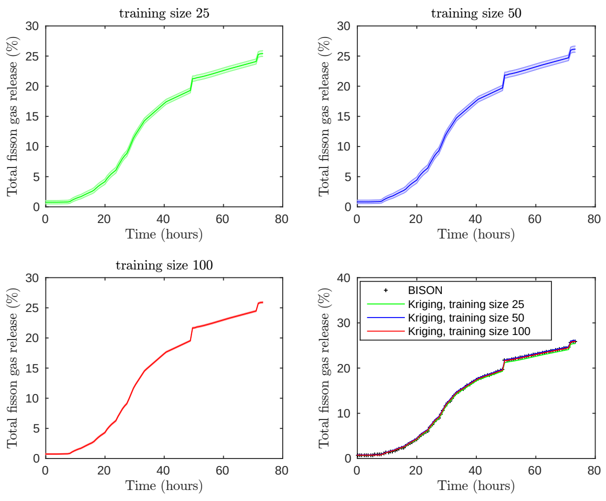


Fig. 8.15 Prediction at nominal values of inputs using GP metamodels built with different number of training samples.

2. Figure 8.14 compares the constant, linear and quadratic trend functions, all with Matérn 5/2 correlation kernel and 100 training samples. No obvious difference can be observed. All the trend functions have good predictions and small MSEs. In fact ordinary GP that uses constant trend functions is the most popular version of GP.

3. Figure 8.15 compares GP predictions that are built with different number of training samples, all with Matérn 5/2 correlation kernel and constant trend function. All the GP metamodels have similar predictions, while the MSEs decrease with more training sampled added. This is expected because the MSE decreases when an untried location

gets closer to training samples. More training samples in the input domain means that any untried location has a high opportunity to have a training sample nearby.

8.4.2 Validating the GP Metamodels

For the validation of GP metamodels built with different number of training samples, we first calculated the Predictivity Coefficient Q_2 for CV to get a general idea of their accuracy. The results are shown in Table 8.3. All the four cases have very high accuracy for the first PC score, while for the second PC score the GP metamodel built with 25 training samples has a bad performance. Realizing that it appears unnecessary to use 200 samples to train the GP metamodels, we use this sample set as a "validation set" for more robust validation.

Table 8.3 Predictivity Coefficients for CV using different number of training samples

Training sample size	First score	Second score
25	0.9804	0.5578
50	0.9720	0.8469
100	0.9895	0.9789
200	0.9921	0.9874

We followed the validation process used in [9] [69]. Figure 8.16 shows the comparison of first and second PC scores from BISON simulations (after projection onto the PC subspace) and GP metamodel predictions. Accurate GP metamodels should have predictions that fall on the diagonal lines. For the first PC score, GP emulators built with different number of training samples have similar accuracy, while for the second PC score, GP emulators built with more samples produce predictions that are more concentrated to the diagonal line. These observations are consistent with the results shown in Table 8.3.

We also calculated the standardized residuals, which is the differences between BISON simulation and GP prediction divided by the corresponding standard deviations of GP predictions.

$$\frac{y^M(\mathbf{x}) - \hat{y}^M(\mathbf{x})}{\sqrt{\mathrm{MSE}\left[\hat{y}^M(\mathbf{x})\right]}} \tag{8.5}$$

The GP metamodel is approximately 99.7% confident that the standardized residuals should lie within the interval $[-3,3]$. Figure 8.17 demonstrates the standardized residuals for first and second PC scores with GP metamodels trained by 25, 50 and 100 LHS samples. The x-axis is the predicted values for PC scores. Clearly the standardized residuals from GP metamodels trained with 25 and 50 samples have many points fall above 3 or below -3. The GP metamodel

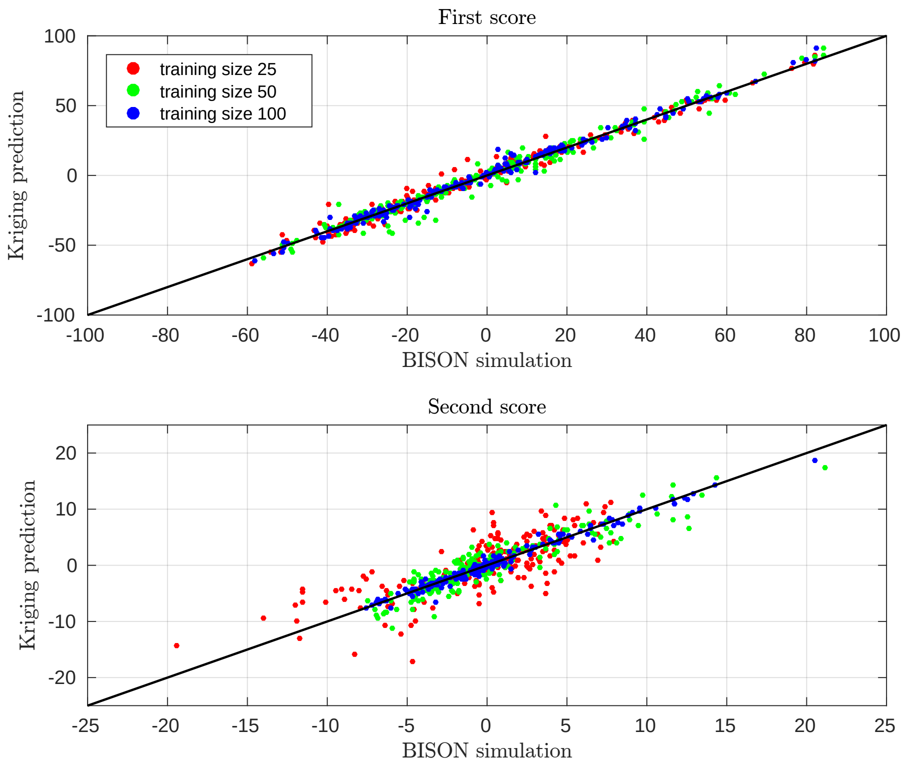


Fig. 8.16 Comparison of first and second PC scores from BISON simulation and GP metamodel prediction.

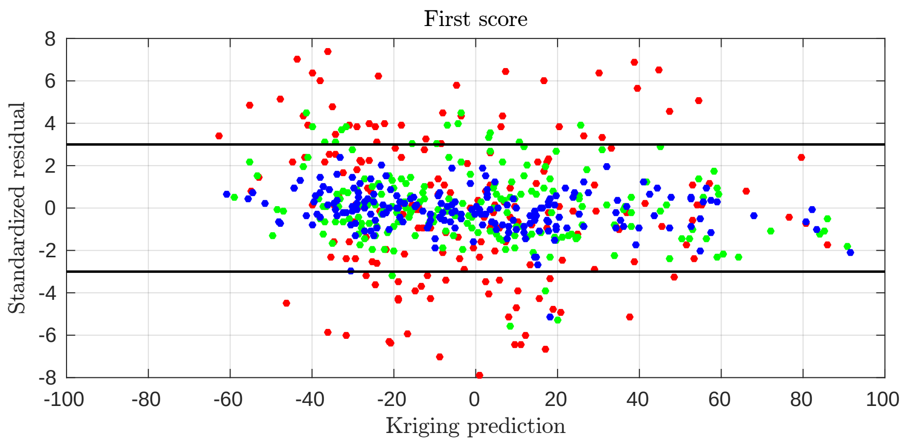
built with 100 training samples have over 99.7% points that stay within $[-3,3]$, suggesting that this metamodel is valid.

Figure 8.18 shows the Q-Q plot, which is the quantiles of standardized residuals versus the quantiles from random samples of the same size from a standard normal distribution. The strong linearity suggests that the standardized residuals are close to being normally distributed. However, we do have a few outliers that make the points deviate from linear line.

8.5 Results for Inverse UQ

8.5.1 Pre-processing the Time Series Measurement Data

After performing the dimension reduction using PCA, building and validating the GP metamodels, we are ready to apply the metamodels in the inverse UQ process. Note that the GP metamodels are built for the responses in the reduced space, i.e., the first and second PC scores. During the inverse UQ process, we have two options to use the measurement data.



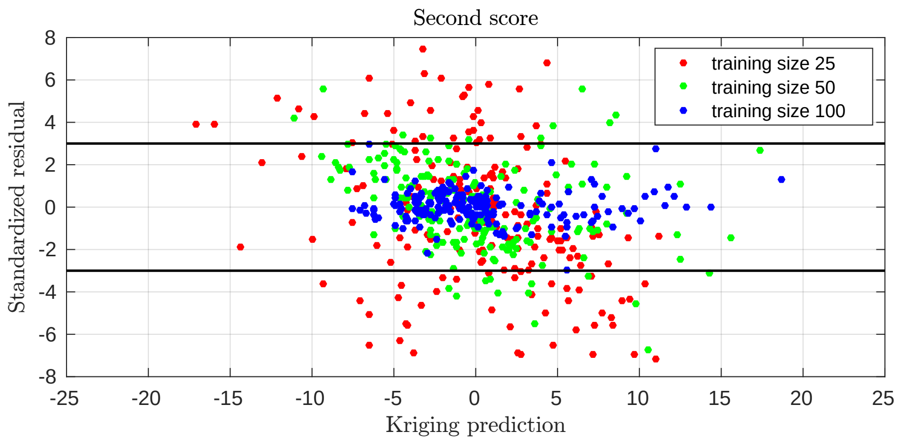


Fig. 8.17 Standardized residuals for first and second PC scores.

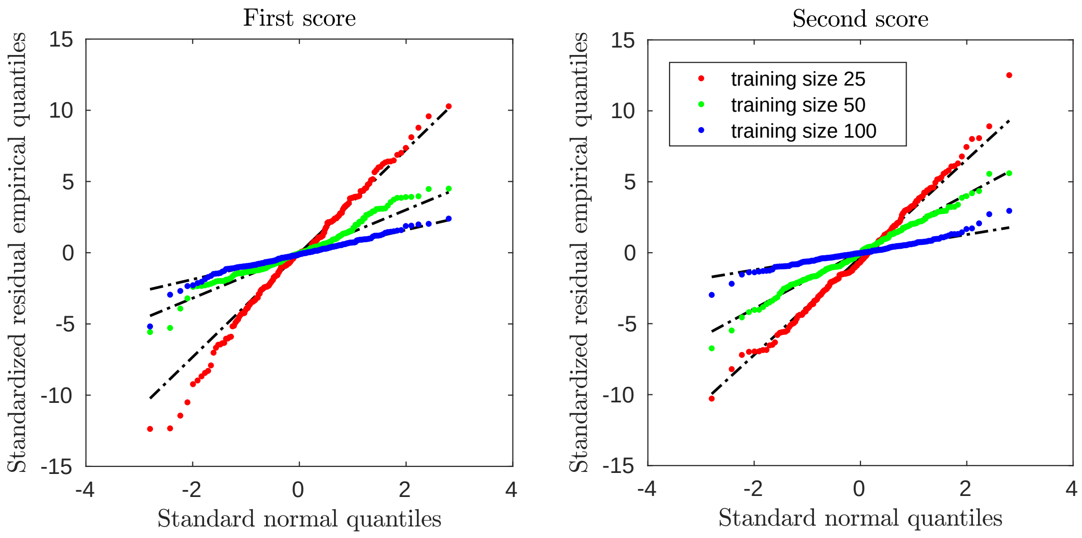


Fig. 8.18 Q-Q plot of the standardized residuals for first and second PC scores.

1. Every vector of GP predictions of the first two PC scores can be transformed back to the original space. Then the Risø-AN3 FGR measurement data will enter the likelihood function directly to decide the acceptance of this MCMC sample. This option is shown by step 2 in Figure 8.6.

2. In another option, instead of transforming the prediction vector to the original space, we transform the Risø-AN3 experimental data to the PC subspace. During MCMC sampling we will use the reduced representation of the experimental data. This option is shown by step 3 in Figure 8.6.

In the current study we chose the second option. Define a $p \times 1$ vector $\mathbf{d}$ which represents the FGR time series in the original space. By centering this vector we get $\mathbf{d}_{\text{centered}} = \mathbf{d} - \mathbf{u}$. Projecting this centered vector onto the PC subspace:

$$\mathbf{P}^{*}\mathbf{d}_{\text{centered}} = \mathbf{g}^{*} = [g_1^{*}, g_2^{*}]^{\top} \tag{8.6}$$

where g_1^{*} and g_2^{*} are the PC scores for the time series data. Because the loading matrix $\mathbf{P}^{*}$ and mean vector $\mathbf{u}$ are calculated using ensemble of BISON executions, projecting $\mathbf{g}^{*}$ back to the original space will results in a FGR time series that has a **similar shape** with BISON simulations. The backward projection is defined as:

$$\mathbf{P}^{*\top}\mathbf{g}^{*} + \mathbf{u} = \mathbf{p}_1^{*\top} g_1^{*} + \mathbf{p}_2^{*\top} g_2^{*} + \mathbf{u} \tag{8.7}$$

Where $\mathbf{p}_1^{*}$ and $\mathbf{p}_2^{*}$ are the first and second PCs corresponding to the rows of $\mathbf{P}^{*}$. Figure 8.19 shows the comparison of original measurement data, smoothed data and reconstructed data using first, second and both PC scores. Due to the complexity and strong-non-linearity of FGR phenomenon and limited representation in current BISON FGR model, it is impossible to find a set of input parameters to make BISON prediction and Risø-AN3 measurements agree well. The reason to choose the second option is that the time series of the reconstructed data using both PCs is the "best match" for the original measurement data that we can find for BISON simulation.

We have defined the covariance matrix for the measurement error as $\mathbf{\Sigma}_{\varepsilon}$. If we assume independence among all the measurements, $\mathbf{\Sigma}_{\varepsilon}$ is a $p \times p$ diagonal matrix with measurement noises as diagonal entries $\mathbf{\Sigma}_{\varepsilon} = \sigma_{\varepsilon}^{2}\mathbf{I}$. In the current we assume 10% measurement error as suggested by BISON FGR developers [108]. Recall that in Chapter 2 Section 2.1 we mentioned the *code/interpolation uncertainty*, which is caused by using metamodels to replace the full model. Statistical models like GP emulator provide estimation of the MSE associated with each prediction, making it possible to account for this source of uncertainty. Define $\mathbf{\Sigma}_{\text{k}}$ as

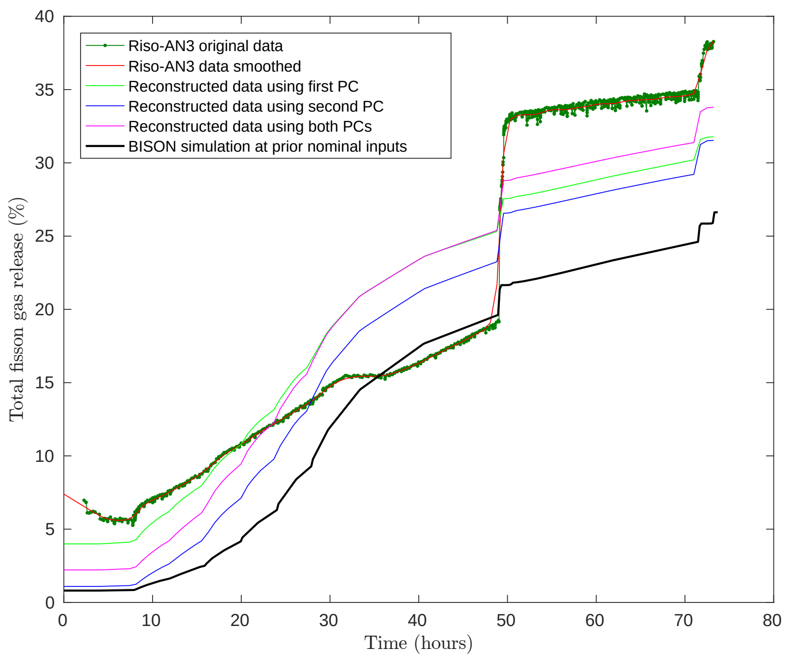


Fig. 8.19 FGR evolution: a comparison of original measurement data, smoothed data and reconstructed data after being projected onto the PC subspace.

the covariance matrix for the PC score predictions by GP, which is also a $p^* \times p^*$ diagonal matrix with the MSE for each PC score as the diagonal entries. The covariance matrix of the likelihood function in the reduced space is defined below, where the first term represents the transformation of the measurement error from the original space to the reduced space:

$$\mathbf{\Sigma} = \mathbf{P}^* \mathbf{\Sigma}_\varepsilon \mathbf{P}^{*\top} + \mathbf{\Sigma}_\mathrm{k} \tag{8.8}$$

8.5.2 Global Sensitivity Analysis using GP Metamodels

The GP metamodels can be evaluated very fast, making Monte Carlo sampling for global SA possible. We use the validated GP metamodels to calculate the Sobol' indices using the "'D-3' method listed in [52]. Figure 8.20 shows the calculated Sobol' indices for PC score 1 and score 2. It can be seen that `temperature_scalef`, `grainradius_scalef` and `igdiffcoeff_scalef` are significant input parameters, while `resolutionp_scalef` and `gbdiffcoeff_scalef` are relatively trivial.

8.5.3 Investigation of MCMC Samples

We applied the adaptive MCMC sampling approach Algorithm 3 to generate 100,000 samples. This number is usually more than enough for a Markov chain. We used the UQLab MATLAB

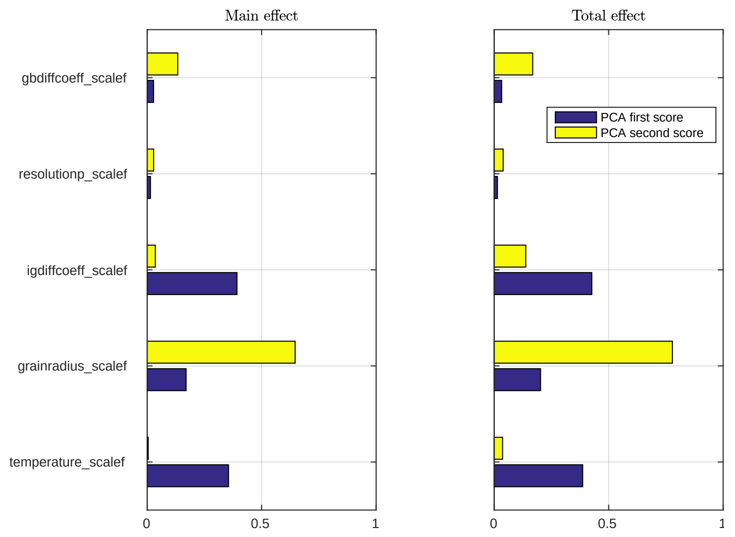


Fig. 8.20 Sobol' indices calculated with GP metamodels.

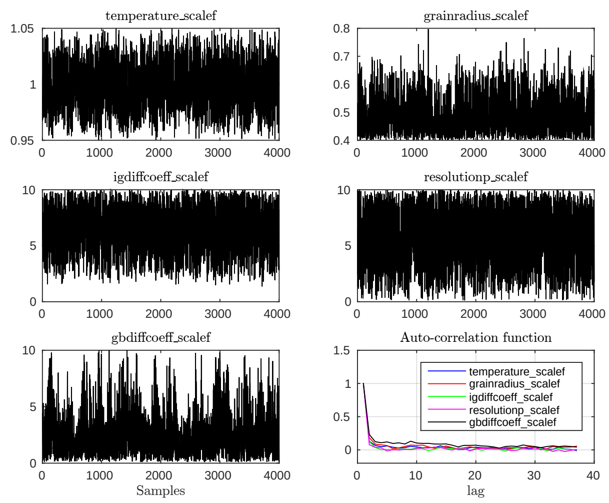


Fig. 8.21 Mixing and auto-correlation functions of the MCMC samples.

package [83] and it takes about 6.5 minutes to generate all the samples with one processor, which would otherwise take 7291 days by running BISON full model with 32 processors. After discarding the first 20,000 samples for burnin, we keep only every 10$^{\text{th}}$ of the remaining samples for thinning of the chain. We investigated the trace plots and decay of auto-correlations to determine the convergence of chain, which are shown in Figure 8.21. Very good mixing can be identified for all the input parameters and the fast decay of the auto-correlations suggest nearly independence.

Table 8.4 shows the statistical moments and 95% Credible Intervals (CIs) for the five input parameters. The 95% CIs are calculated using Highest Posterior Density (HPD) intervals, which are the narrowest intervals including the mode. Only the scale factor for temperature

Table 8.4 Posterior statistical moments and 95% CIs for five uncertain input parameters.

Parameter	Mean values	Standard deviations	Modes	95% CI
`temperature_scalef`	0.9985	0.0212	0.9985	[0.9577, 1.0367]
`grainradius_scalef`	0.4903	0.0660	0.4512	[0.4001, 0.6213]
`igdiffcoeff_scalef`	6.4707	2.0521	7.3210	[2.7721, 9.2956]
`resolutionp_scalef`	5.7717	2.4799	6.8576	[1.4260, 9.3975]
`gbdiffcoeff_scalef`	2.4686	1.8652	1.0716	[0.1806, 6.4401]

is close to the nominal value 1.0. Large standard deviations are found for the last three scale factors.

Table 8.5 Correlation matrix based on MCMC samples

	temperature_scalef	grainradius_scalef	igdiffcoeff_scalef	resolutionp_scalef	gbdiffcoeff_scalef
`temperature_scalef`	1.00				
`grainradius_scalef`	0.36	1.00			
`igdiffcoeff_scalef`	-0.68	0.12	1.00		
`resolutionp_scalef`	-0.22	0.11	-0.04	1.00	
`gbdiffcoeff_scalef`	-0.73	-0.54	0.16	0.12	1.00

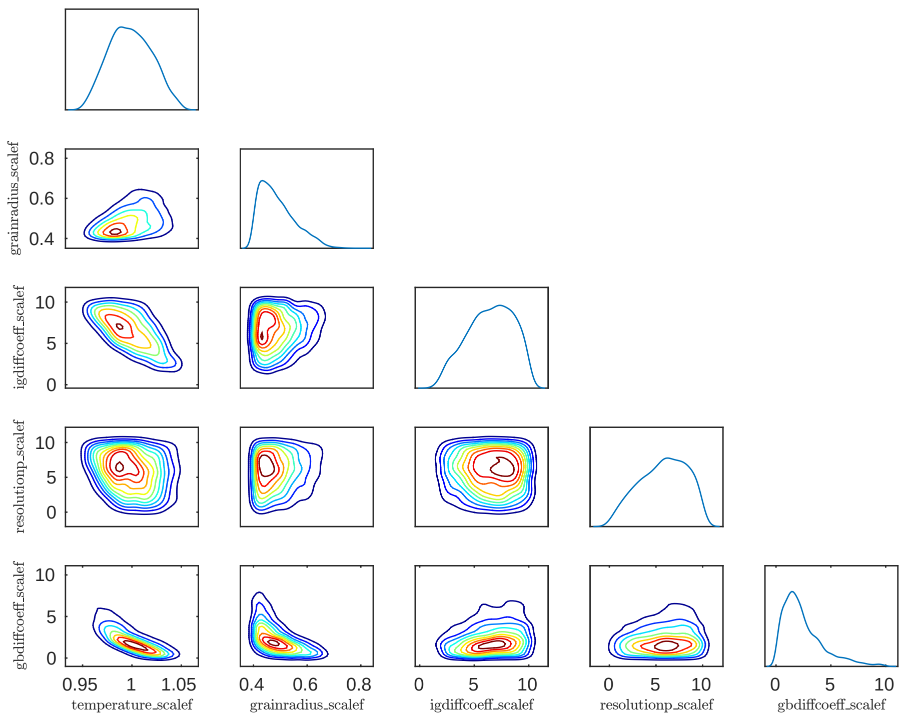


Fig. 8.22 Posterior marginal and joint distributions for the five uncertain input parameters based on MCMC samples.

Table 8.5 presents the correlation matrix. All the last four scale factors have certain degree of correlation with the temperature scale factor. This is expected as they are essentially more or less dependent on temperature [154]. Figure 8.22 shows the marginal and pair-wise joint distributions of the five uncertain input parameters. The pair-wise joint densities clearly show that some parameters are correlated, which is consistent with results in Table 8.5.

8.5.4 Fitted Posterior Distributions

To make these results more applicable to future analysis, we need to fit posterior samples to well-known distributions, such as Normal distribution, so that they will be more easily sampled. Figure 8.23 and Table 8.6 show the fitted distribution for each uncertain scale factors and the parameters associated with each distribution, i.e. mean (μ) and standard deviation (σ) for normal distribution, shape (α) and scale (β) parameter for Gamma distribution, location (μ), scale (σ), and shape parameter (k) for generalized extreme value distribution. All the fitted distributions are accepted by Kolmogorov-Smirnov test at the 5% significance level. Figure 8.24 shows that good agreement can be achieved between the empirical CDF based on MCMC samples and fitted CDF for every parameter.

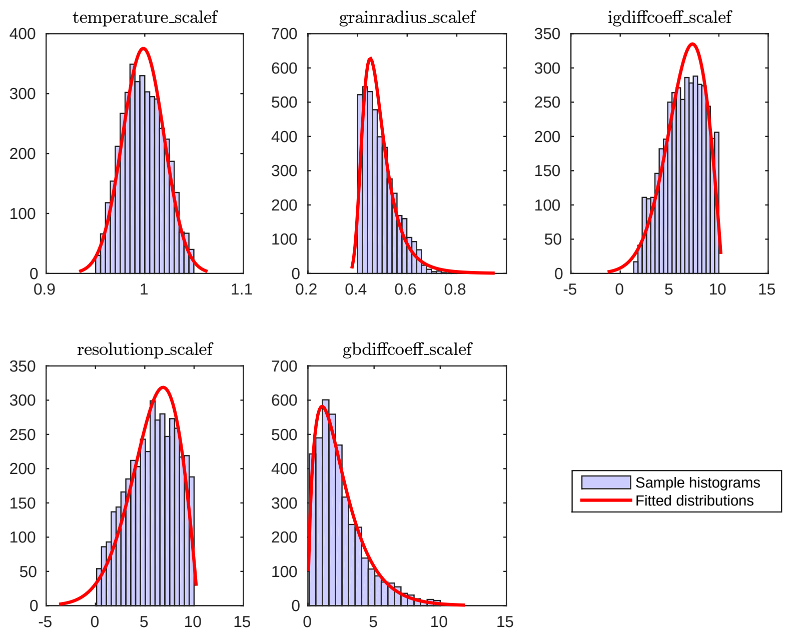


Fig. 8.23 Fitted posterior probability densities

Table 8.6 Fitted distributions for each uncertain input parameters

Parameter	Distribution type	Distribution parameters
`temperature_scalef`	Normal	$\mu = 0.9985, \sigma = 0.0211$
`grainradius_scalef`	Generalized Extreme Value	$\mu = 0.1290, \sigma = 0.0469, k = 0.4569$
`igdiffcoeff_scalef`	Generalized Extreme Value	$\mu = -0.5123, \sigma = 2.2303, k = 5.9814$
`resolutionp_scalef`	Generalized Extreme Value	$\mu = -0.5237, \sigma = 2.7053, k = 5.1948$
`gbdiffcoeff_scalef`	Gamma	$\alpha = 1.7671, \beta = 1.3940$

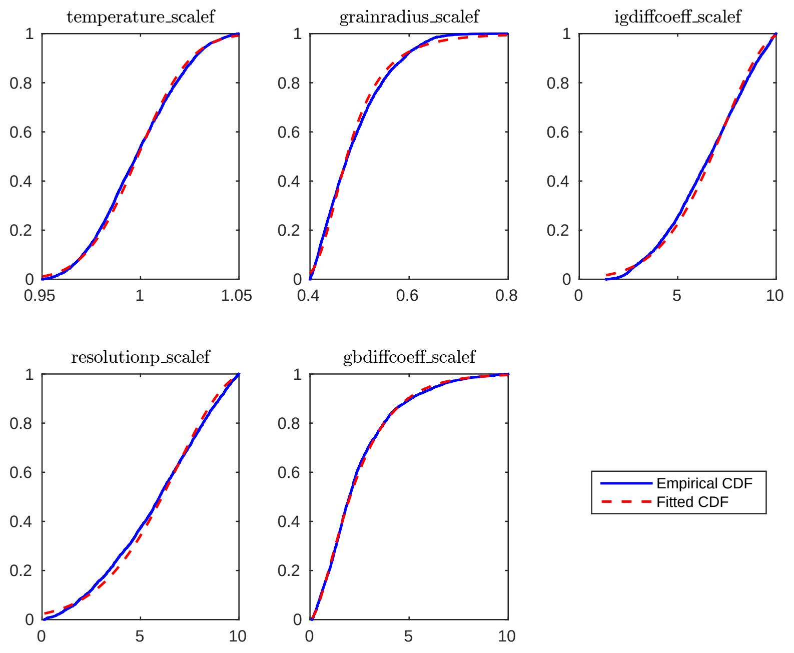


Fig. 8.24 Comparison of empirical CDFs from MCMC samples and fitted CDFs.

8.6 Results for Forward UQ based on Posteriors

Posterior probabilities are the degree of belief about possible values of the uncertain input parameters after observing the experimental data. From Section 8.5 we have achieved the posterior distributions. To show that these updated uncertainty information is indeed consistent with Risø-AN3 FGR time series data, we performed a forward uncertainty propagation for BISON FGR simulation using posterior distributions. We generated a new set of 200 Monte Carlo samples using max-min LHS design, based on the fitted distributions from Section 8.5. This number of samples is an arbitrary choice and we have demonstrated in Section 8.3 that it is enough to provide a good sense of the lower and upper bounds. Note that these samples are generated by considering the correlation of the five input scale factors as shown in Table 8.5. Figure 8.25 shows the generated samples.

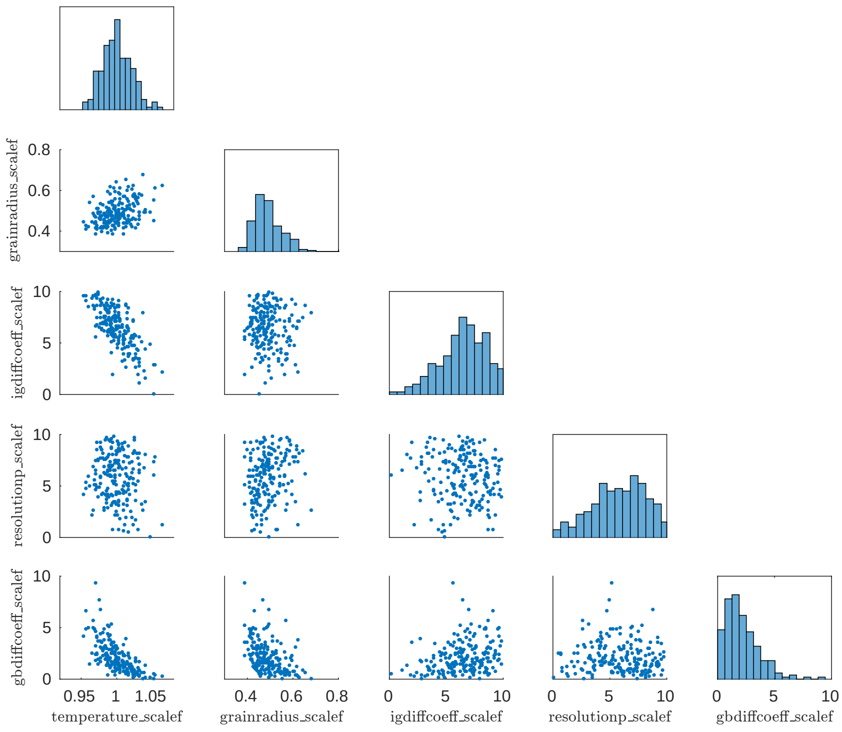


Fig. 8.25 Samples generated based on fitted posterior distributions.

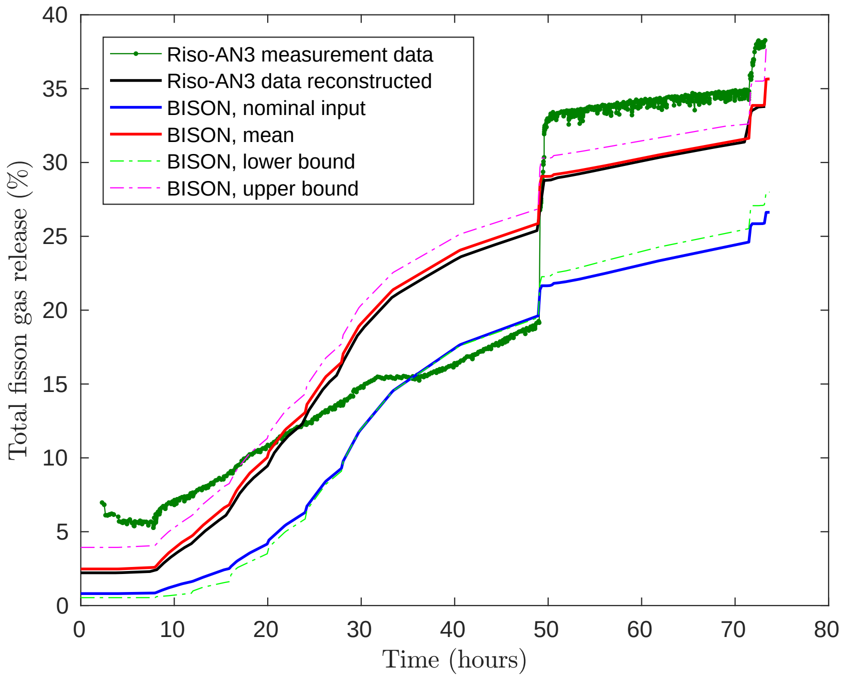


Fig. 8.26 Comparison of FGR time series from Risø-AN3 benchmark original measurement data, reconstructed measurement data, BISON simulation with nominal inputs and BISON simulation with posterior samples.

Figure 8.26 shows the comparison of FGR time series from (1) Risø-AN3 measurement data; (2) Risø-AN3 data reconstructed using both PCs; (3) BISON simulation at prior nominal inputs (1.0); (4) Mean, lower and upper bounds of BISON simulations at 200 LHS samples generated according to posterior distributions. Note that we also run GP metamodel at these samples and very close statistical results (mean, lower and upper bounds) are observed to BISON full model simulations. In Figure 8.26 we only show BISON full model simulation results. It can be observed that:

1. The mean of BISON simulations are very close to the reconstructed Risø-AN3 time series data. In Section 8.5 we have pointed out that such reconstructed time series data is the best we can do under two constraints: (1) it is consistent with original measurement data; (2) it has a shape that is similar with BISON simulations.

2. The mean of BISON simulations according to posterior distributions demonstrates a much better agreement with measurement data, compared with the simulation using prior nominal inputs. Furthermore, BISON simulations at mean values and modes that are listed in Table 8.4 are very close to the means of BISON simulations at these max-min LHS samples. Therefore, if we use the posterior mean values as new nominal input values, BISON FGR prediction will be close with the red curve (labeled "BISON, mean") in Figure 8.26.

3. The lower and upper bounds do not envelop the original Risø-AN3 measurement data anymore. However, this is not a constraint for posterior distributions.

Figure 8.27 and Figure 8.28 show the fuel centerline temperature and rod internal pressure based on the posterior respectively. No obvious change can be observed for fuel centerline temperature. Even though FGR and fuel temperature form a positive feedback as explained in [154], the increased FGR does not cause the fuel temperature to increase because the transient is too short.

The agreement of rod internal pressure from BISON simulation based on the posterior and Risø-AN3 benchmark gets worse compared with prior nominal values. It is expected to have a higher rod internal pressure when the released fission gases increase Note that BISON already over-predicts the rod internal pressure with prior nominal values. Therefore, possible explanations can be that (1) the rod internal pressure measurement is severely under-estimated; (2) there are some missing or insufficient physics in BISON simulation of rod internal pressure, which causes the over-estimation of the pressure.

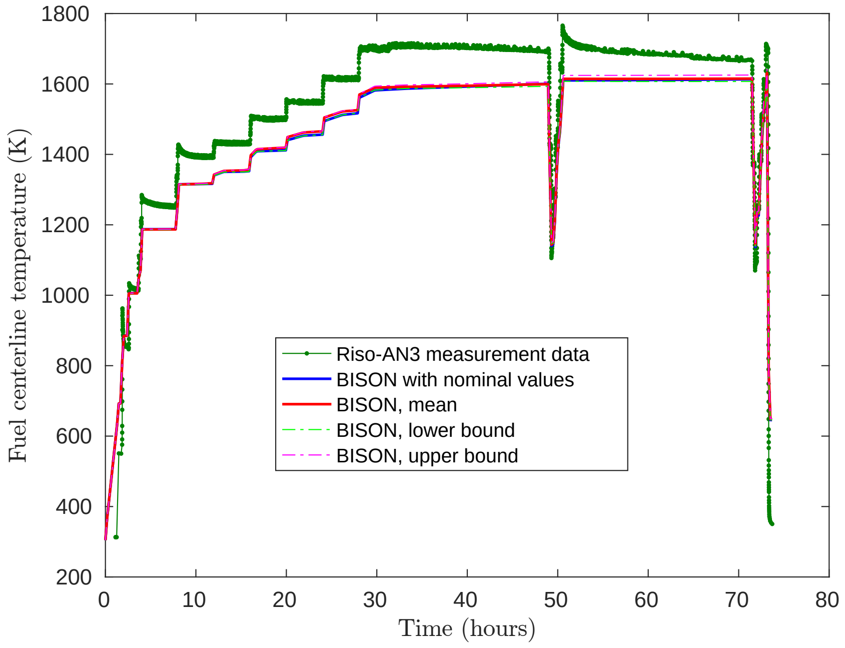


Fig. 8.27 Comparison of fuel centerline temperature from Risø-AN3 benchmark, BISON simulation with nominal inputs and BISON simulation with posterior samples.

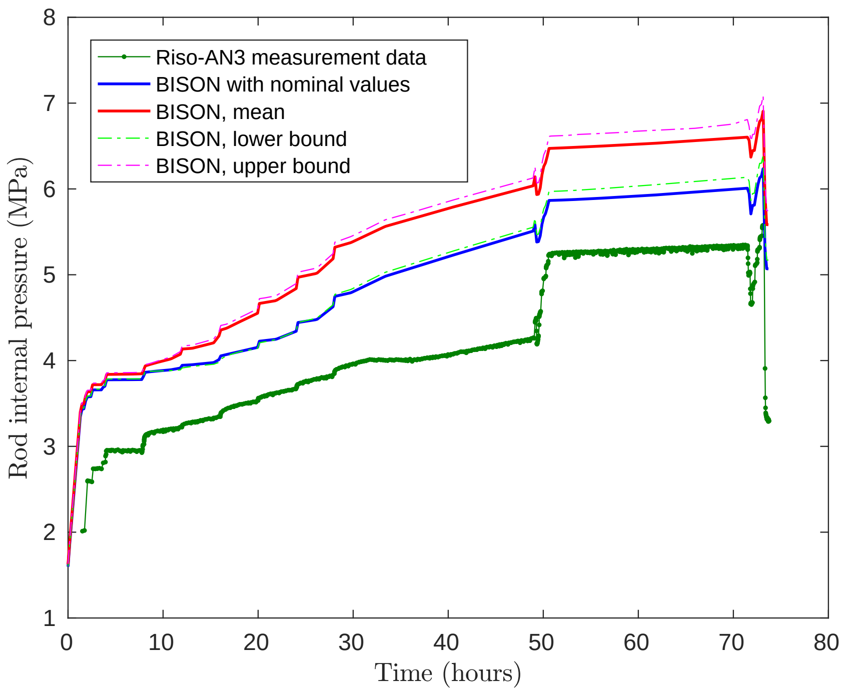


Fig. 8.28 Comparison of rod internal pressure from Risø-AN3 benchmark, BISON simulation with nominal inputs and BISON simulation with posterior samples.

8.7 Discussions

Some limitations of the applied approach and results in this chapter are:

1. It is certain that the disagreement between BISON FGR simulations and experiment measurements are caused not just by the uncertainty in the five scale factors, but also model discrepancy arising from missing or insufficient physics and numerical approximations. For example, as claimed by the FGR model developers [106] [146], an example of insufficient physics is the burst release of fission gases accumulated at the grain boundaries during a sudden power drop. Due to the complexity of the phenomena, the present model is BISON is useful but limited.

2. Even though we have shown that over 99.7% percent of the total variation can be accounted for by the first two principal dimensions, it is unclear how much extra uncertainties will be introduced by discarding all the rest principal dimensions. The authors in [144] consider data loss by using Gaussian multipliers of the discarded basis vectors. But in the current research we did not do it as the inverse UQ process was done in the reduced space rather than the original space.

3. We did not consider model discrepancy as there was not enough information available. We used only one set of time series measurement data in the present work. There are several other benchmarks that have FGR measured [62] [108]. However, only very few of them have on-line, time-dependent measurement of FGR. These measurement data are not enough to train a statistical model for the model discrepancy.

4. Though we have fitted BISON FGR model to Risø-AN3 time series measurement data, it is questionable if the confidence in FGR predictions can be improved when BISON is applied to different fuel designs or irradiation conditions.

Chapter 9

CONCLUSIONS AND FUTURE WORK

9.1 Conclusions

Mathematical modeling and computer simulations have long been the central technical topics in practically all branches of science and technology. They are naturally affected by a relatively large amount of uncertainties coming from numerical algorithms, model parameters, initial and boundary conditions, user effects, etc. Confidence in modeling and simulation must be critically assessed which requires model V&V. Forward UQ plays a vital role in the validation process. However, Uncertainties pertaining to input parameters are ubiquitous because: (1) even the best computer models are reduced representations of the real phenomena; (2) our knowledge of the underlying physics is incomplete. Identification and characterization of input parameter uncertainties are essentially the beginning step for computer model uncertainty analysis.

However, previously "expert judgment" or "user self-evaluation" haven been widely used to address the "lack of input uncertainty information" issue for random input parameters. The thesis seeks to replace such ad-hoc specification of input uncertainties by inverse UQ process, which adopts a Bayesian setting and uses MCMC sampling to explore the posterior distributions. Inverse UQ aims to quantify the uncertainty in input parameters such that the discrepancies between code output and observed experimental data can be minimized. Because usually hundreds of thousands of samples are required during MCMC sampling, accurate and fast-running metamodels are developed to replace the full model especially when the full model is computationally prohibitive. We investigated the stochastic spectral techniques and GP modeling to build metamodels.

In Chapter 5, inverse UQ is applied to a simplified nuclear reactor simulation problem, the PRKE coupled with lumped parameter TH feedback model. Three input parameters are considered random, i.e., the external reactivity insertion, the Doppler temperature coefficient and the coolant temperature coefficient. Generalized PCE approximates model QoIs as polynomial functions of random input parameters. Such functions have demonstrated that they can successfully represent the stochastic nature of the outputs in terms of their mean values and variances. The PCE essentially constitutes a metamodel of the original problem and can be used as surrogates during the inverse UQ process. Based on the results from inverse UQ, we found a very good agreement of the statistical moments and PDFs of posteriors between full simulation and using PCE metamodels (with enough high order).

When PCE are used as metamodels, only polynomial evaluations are required for MCMC sampling, which reduces the simulation time by 2-3 orders of magnitude compared to direct simulation of the full model, even for a simple model considered here. Furthermore, once the PCE metamodels are ready (modes calculated), they can be used without modification when new experimental data is available. It has also been demonstrated that different priors will result in close posteriors. Based on the validation results, it can be concluded that the inverse UQ with Bayesian analysis can update prior uncertainties such that the simulation results will agree better with experimental data and simulation uncertainty envelops the measurement data.

In Chapter 6, we applied inverse UQ under the Bayesian framework using a SGSC metamodel to quantify the uncertainties in TRACE physical model parameters based on BFBT benchmark steady-state void fraction data. Two global sensitivity analysis methods, Sobol' indices and correlation coefficients, were used to identify the important TRACE physical model parameters for the BFBT benchmark. Starting with 36 possible physical models, 8 physical models were selected using centered parameter study, and their Sobol' indices and PCC/SRCC were calculated to evaluate their significance with regards to the void fraction data. Based on the two global sensitivity measures, 5 input parameters were selected upon which the inverse UQ study was performed.

SGSC metamodels were constructed for TRACE based on selected BFBT test cases. MCMC samples provided statistical information for the 5 input parameters (i.e. the mean values, standard deviations, and PDFs). Lastly, Gaussian and Gamma distributions were fitted for 5 physical model parameters and the parameters for these distributions were reported. This research addresses the problem of lacking uncertainty information about TRACE physical input parameters, which has been often ignored or described using expert opinion or personal judgment in prior uncertainty and sensitivity analysis work. The results of inverse UQ of TRACE physical model parameters are critical for future uncertainty and sensitivity study of TRACE code for use in nuclear reactor system design and safety analysis.

In Chapter 7, the same problem with Chapter 6 was analyzed with the improved modular Bayesian approach developed in Chapter 2 Section 2.3. In Chapter 2, We provided a detailed introduction and comparison of the full and modular Bayesian approach for inverse UQ. Inverse UQ with both full and modular Bayesian are fully data-driven. Therefore, these methods should be used with great caution. The extrapolation of the model discrepancy term apparently depends on the error/uncertainty structure in the validation/prediction domain. As the model discrepancy term is trained based on experimental data in the inverse UQ domain, its extrapolation in the validation/prediction domain is a sophisticated and unresolved issue. We proposed an improved modular Bayesian approach that can avoid extrapolating the model discrepancy that is learnt from the inverse UQ domain to the validation/prediction domain. The improved approach is organized in a structure such that the posteriors achieved with data in inverse UQ domain is informed by data in the validation domain. Therefore, over-fitting can be avoided while extrapolation is not required.

A sequential approach was also developed for test source allocation (TSA) for inverse UQ and validation. This sequential TSA methodology first select tests for validation that has a full coverage of the test domain to avoid extrapolation of model discrepancy term when evaluated at input setting of tests for inverse UQ. Then it select tests that tend to reside in the unfilled zones of the test domain for inverse UQ, so that inverse UQ can extract the most information for posteriors of calibration parameters using only a relatively small number of tests.

The model discrepancy term is described with a GP emulator `GPbias`. Numerical tests have demonstrated that such treatment of model discrepancy can avoid over-fitting. Furthermore, we constructed a fast-running and accurate GP emulator `GPcode` to replace TRACE full model during MCMC sampling. The number of TRACE runs is reduced by about two orders of magnitude (from 50,000 to less than 500), and the MCMC sampling time is reduced from about 24 core-days to 37 core-minutes.

In Chapter 8, inverse UQ was applied to fuel performance code BISON FGR model based on Risø-AN3 FGR time series measurement data. Kriging metamodels were applied to greatly alleviate the computer burden during MCMC sampling. Because the time series data corresponds to high-dimensional outputs, we performed dimension reduction using PCA. In this way we bypassed the issues related to building a Kriging metamodel and built a few independent metamodels for the PC scores instead. Unlike previous studies that reconstruct Kriging predictions to the original space by inverse projection, we performed inverse UQ on the PC subspace. The original FGR time series data is projected onto the PC subspace as "new experimental data". Such an approach is chosen because BISON FGR simulation differs severely in shape with measurement data. It is shown that the reconstructed time series data is the best we can do under two constraints: (1) it is consistent with original measurement data; (2) it has a shape that is similar with BISON simulations.

A forward uncertainty propagation based on the fitted distributions shows that the agreement between BISON simulation and Risø-AN3 time series measurement data is greatly improved. We provided uncertain distributions for the five random scale factors in BISON FGR model by a rigorous inverse UQ process, which can be used to replace the relatively arbitrary expert specifications for future uncertain propagation.

9.2 Future Work

The development of advanced computational platforms allows the simulation of sophisticated multi-physical phenomena within nuclear reactors in a coupled fashion. Current physical phenomenon involved in nuclear reactor modeling include neutron transport, thermal-hydraulics, fuel performance, and coolant chemistry. There are some inherent difficulties associated with these modules that pose immense challenges to the nuclear-related VVUQ study:

1. The experimental observations may be noisy, time-dependent and correlated. The application of such data can result in a computationally intractable problem if each of the highly correlated observations is to be considered individually. For example, it is inefficient to build metamodels for observations at each time step. We have demonstrated the application of PCA for dimension reduction when performing inverse UQ for fuel performance code fission gas release model. Dimension reduction techniques should be further improved for dealing with more complicated data. The common Bayesian framework (i.e. assuming i.i.d. additive Gaussian noise) will also be adapted for such specific requirements.

2. Even though the state-of-the-art nuclear reactor simulators have achieved tremendous successes in representing real phenomena, as an approximation to reality they are still expected to be biased, the form and magnitude of which is not necessarily known. As a solution, a statistical model based on GP should be employed to represent the model discrepancy/bias term [4] [13] [168]. Such effort is essential to avoid over-fitting the calibration to a specific dataset, thus rendering the results not applicable in new dissimilar conditions.

3. Given the limited amount of multi-physics data available in the nuclear engineering field, Bayesian hierarchical modeling [110] [140] should be used, which allows the incorporation of multiple data sources for solving stochastic inverse problems. For example, temperature, void fraction and pressure drop will be folded together to provide a holistic and multi-physics calibration approach.

4. Results from inverse UQ and validation should be integrated to improve the computer model predictive capability. Note that since both processed are fully data-drive, extrapolation of information should be avoided, e.g. the model discrepancy term.

Appendix A

Classical Orthogonal Polynomials

In this appendix, some classical continuous orthogonal polynomials are introduced, including Hermite, Laguerre, Legendre and Jacobi polynomials.

A.1 Hermite Polynomials

Definition and differential equation:

There are two different but closely related definition of Hermite polynomials, namely, the "probabilists' Hermite polynomials" and the "physicists' Hermite polynomials".

The "probabilists' Hermite polynomials" are given by:

$$He_n(x) = (-1)^n e^{\frac{x^2}{2}} \frac{d^n}{dx^n} e^{-\frac{x^2}{2}} = \left(x - \frac{d}{dx}\right)^n \cdot 1 \tag{A.1}$$

While the "physicists' Hermite polynomials" are given by:

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2} = \left(2x - \frac{d}{dx}\right)^n \cdot 1 \tag{A.2}$$

Note that each one of the two definitions are actually a rescaling of the other, with the following relations:

$$\begin{aligned} H_n(x) &= 2^{\frac{n}{2}} \cdot He_n(\sqrt{2}x) \\ He_n(x) &= 2^{-\frac{n}{2}} \cdot H_n(\frac{1}{\sqrt{2}}x) \end{aligned} \tag{A.3}$$

The Hermite Polynomials are solutions of the following differential equations:

$$\begin{aligned} y'' - xy' + ny = 0, &\quad n \in \mathbb{N}, \quad \text{for } He_n(x) \\ y'' - 2xy' + 2ny = 0, &\quad n \in \mathbb{N}, \quad \text{for } H_n(x) \end{aligned} \tag{A.4}$$

Here we will only look at the "physicists' Hermite polynomials" $He_n(x)$.

Recurrence relation:

The Hermite polynomials can be generated in practice by the following recurrence relations:

$$He_{n+1}(x) = xHe_n(x) - He'_n(x), \quad \text{for } n = 0, 1, 2, \cdots \tag{A.5}$$

The first few probabilists' Hermite polynomials are given by:

$$\begin{aligned} He_0(x) &= 1 \\ He_1(x) &= x \\ He_2(x) &= x^2 - 1 \\ He_3(x) &= x^3 - 3x \\ He_4(x) &= x^4 - 6x^2 + 3 \\ He_5(x) &= x^5 - 10x^3 + 15x \\ He_6(x) &= x^6 - 15x^4 + 45x^2 - 15 \end{aligned} \tag{A.6}$$

Figure A.1 shows the plot of Hermite polynomials up to order 5.

Orthogonality:

Given Hermite polynomials $He_n(x)$ of degree $n = 0, 1, 2, \cdots$, they are orthogonal with respect to properly chosen weight functions, which are

$$w(x) = e^{-\frac{x^2}{2}} \tag{A.7}$$

Now we have:

$$\int_{-\infty}^{\infty} He_m(x)He_n(x)w(x)dx = h_n^2\delta_{mn} = \sqrt{2\pi}n!\delta_{mn} \tag{A.8}$$

where δ_{mn} is the Kronecker delta.

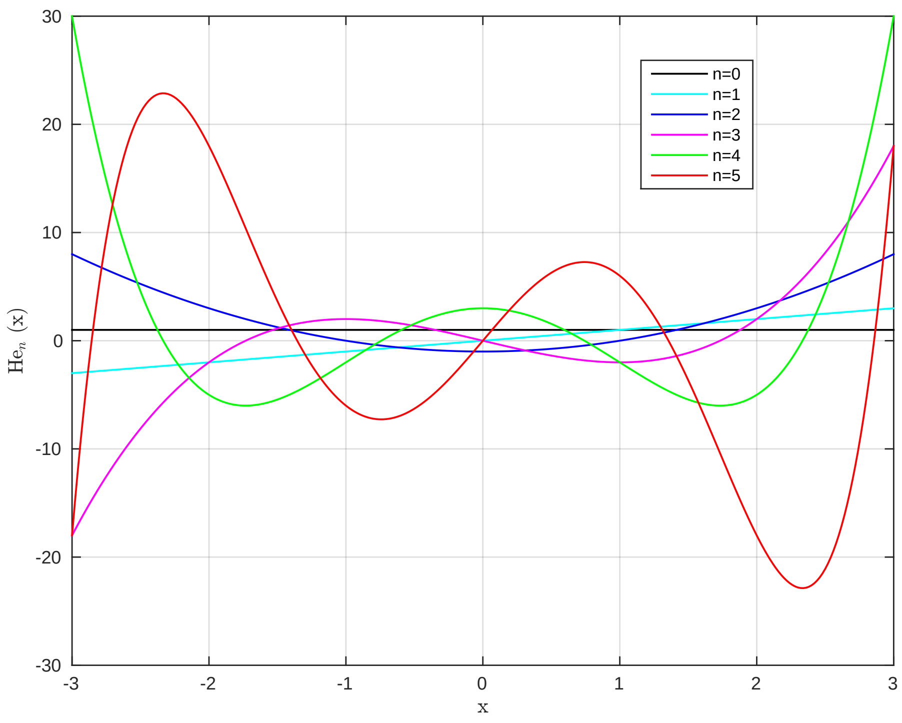


Fig. A.1 Hermite Polynomials (Probabilists')

Since $\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$ is the probability density function (PDF) of the standard normal distribution, the "probabilists' Hermite polynomials", $He_n(x)$ are orthogonal with respect to the PDF of standard normal distribution:

$$\int_{-\infty}^{\infty} He_m(x)He_n(x)\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}dx = n!\delta_{mn} \tag{A.9}$$

A.2 Laguerre Polynomials

Definition and differential equation:

The Laguerre polynomials $L_n(x)$ are solutions of Laguerre's equation:

$$xy'' + (1-x)y' + ny = 0, \quad n \in \mathbb{N} \tag{A.10}$$

If we introduce arbitrary real value α, we get the generalized Laguerre polynomials $L_n^{\alpha}(x)$, or associated Laguerre polynomials from the following differential equation:

$$xy'' + (\alpha+1-x)y' + ny = 0, \quad n \in \mathbb{N} \tag{A.11}$$

The closed form for $L_n(x)$ is:

$$L_n(x) = \sum_{k=0}^{n} \binom{n}{k} \frac{(-1)^k}{k!} x^k \tag{A.12}$$

Recurrence relation:

To generate Laguerre polynomials $L_n(x)$ recursively, use the following recurrence relation:

$$(n+1)L_{n+1}(x) = (2n+1-x)L_n(x) - nL_{n-1}(x), \quad \text{for } n = 1,2,3,\ldots \tag{A.13}$$

For generalized Laguerre polynomials $L_n^\alpha(x)$, the recurrence relation is:

$$(n+1)L_{n+1}^\alpha(x) = (2n+1+\alpha-x)L_n^\alpha(x) - (n+\alpha)L_{n-1}^\alpha(x), \quad \text{for } n = 1,2,3,\ldots \tag{A.14}$$

The first few Laguerre polynomials are:

$$\begin{aligned}
L_0(x) &= 1 \\
L_1(x) &= -x+1 \\
L_2(x) &= \frac{1}{2}(x^2-4x+2) \\
L_3(x) &= \frac{1}{6}(-x^3+9x^2-18x+6) \\
L_4(x) &= \frac{1}{24}(x^4-16x^3+72x^2-96x+24) \\
L_5(x) &= \frac{1}{120}(-x^5+25x^4-200x^3+600x^2-600x+120) \\
L_6(x) &= \frac{1}{720}(x^6-36x^5+450x^4-2400x^3+5400x^2-4320x+720)
\end{aligned} \tag{A.15}$$

Figure A.2 shows the plot of Laguerre polynomials up to order 5.

Orthogonality:

The generalized Laguerre polynomials $L_n^\alpha(x)$ are orthogonal over $[0,\infty]$ with respect to the following weighting function:

$$w(x) = x^\alpha e^{-x} \tag{A.16}$$

$$\int_0^\infty L_m^\alpha(x)L_n^\alpha(x)w(x)dx = h_n^2\delta_{mn} = \frac{\Gamma(n+\alpha+1)}{n!}\delta_{mn} \tag{A.17}$$

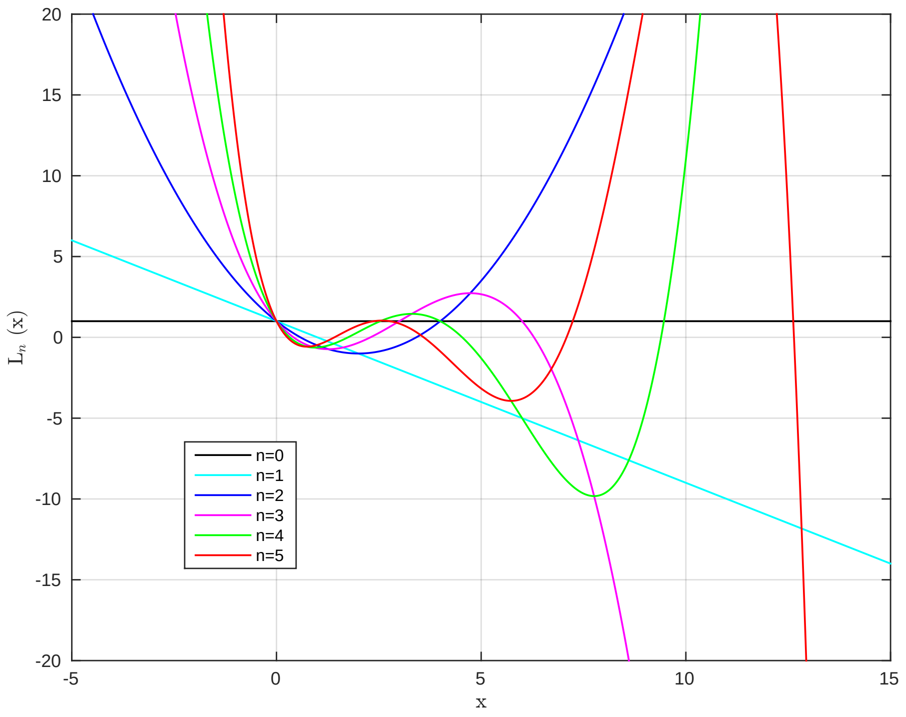


Fig. A.2 Laguerre Polynomials

For the special case when $\alpha = 0$, we will have:

$$\int_0^\infty L_m(x)L_n(x)e^{-x}dx = \frac{\Gamma(n+1)}{n!}\delta_{mn} = \delta_{mn} \tag{A.18}$$

A.3 Jacobi Polynomials

Definition and differential equation:

The Jacobi polynomials $P_n^{(\alpha,\beta)}(x)$, sometimes called hypergeometric polynomials, are the solution of the following differential equation:

$$(1-x^2)y'' + [\beta - \alpha - (\alpha+\beta+2)x]y' + n(n+\alpha+\beta+1)y = 0, \quad n \in \mathbb{N} \tag{A.19}$$

The closed form for $P_n^{(\alpha,\beta)}(x)$ is:

$$P_n^{(\alpha,\beta)}(x) = \frac{\Gamma(n+\alpha+1)}{\Gamma(n+\alpha+\beta+1)n!}\sum_{k=0}^{n}\binom{n}{k}\frac{\Gamma(n+k+\alpha+\beta+1)}{\Gamma(k+\alpha+1)}\left(\frac{x-1}{2}\right)^k \tag{A.20}$$

Another definition is given by the Rodrigues' formula:

$$P_n^{(\alpha,\beta)}(x) = \frac{(-1)^n}{2^n n!}(1-x)^{-\alpha}(1+x)^{-\beta}\frac{d^n}{dx^n}\left\{(1-x)^{\alpha}(1+x)^{\beta}(1-x^2)^n\right\} \tag{A.21}$$

Recurrence relation:

The recurrence relation for the Jacobi polynomials is:

$$\begin{aligned}&2n(n+\alpha+\beta)(2n+\alpha+\beta-2)P_n^{(\alpha,\beta)}(x) = \\ &\quad (2n+\alpha+\beta-1)\left\{(2n+\alpha+\beta)(2n+\alpha+\beta-2)x+\alpha^2-\beta^2\right\}P_{n-1}^{(\alpha,\beta)}(x) - \\ &\qquad 2(n+\alpha-1)(n+\beta-1)(2n+\alpha+\beta)P_{n-2}^{(\alpha,\beta)}(x), \quad \text{for } n = 2,3,4,\ldots\end{aligned} \tag{A.22}$$

Orthogonality:

The Jacobi polynomials $P_n^{(\alpha,\beta)}(x)$ are orthogonal with respect to the following weight function on the interval $[-1,1]$:

$$w(x) = (1-x)^{\alpha}(1+x)^{\beta} \tag{A.23}$$

$$\int_{-1}^{1} P_m^{(\alpha,\beta)}(x)P_n^{(\alpha,\beta)}(x)(1-x)^{\alpha}(1+x)^{\beta}dx = h_n^2\delta_{mn}, \quad \alpha,\beta,\alpha+\beta > -1 \tag{A.24}$$

$$h_n^2 = \frac{2^{\alpha+\beta+1}}{2n+\alpha+\beta+1}\frac{\Gamma(n+\alpha+1)\Gamma(n+\beta+1)}{\Gamma(n+\alpha+\beta+1)n!} \tag{A.25}$$

A.4 Legendre Polynomials

Definition and differential equation:

The Legendre polynomials $P_n(x)$, are special cases of Jacobi polynomials $P_n^{(\alpha,\beta)}(x)$ with $\alpha = \beta = 0$. They are the solution of the following Legendre's differential equation:

$$(1-x^2)y'' - 2xy' + n(n+1)y = 0, \quad n \in \mathbb{N} \tag{A.26}$$

With Rodrigues' formula:

$$P_n(x) = \frac{(-1)^n}{2^n n!}\frac{d^n}{dx^n}\left\{(1-x^2)^n\right\} = \frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n \tag{A.27}$$

Recurrence relation:

The Legendre polynomials $P_n(x)$ have the following recurrence relation:

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x), \quad n = 1,2,3,\ldots \tag{A.28}$$

The first few Laguerre polynomials are:

$$\begin{aligned}
P_0(x) &= 1 \\
P_1(x) &= x \\
P_2(x) &= \frac{1}{2}(3x^2-1) \\
P_3(x) &= \frac{1}{2}(5x^3-3x) \\
P_4(x) &= \frac{1}{8}(35x^4-30x^2+3) \\
P_5(x) &= \frac{1}{8}(63x^5-70x^3+15x) \\
P_6(x) &= \frac{1}{16}(231x^6-315x^4+105x^2-5)
\end{aligned} \tag{A.29}$$

Figure A.3 shows the plot of Legendre polynomials up to order 5.

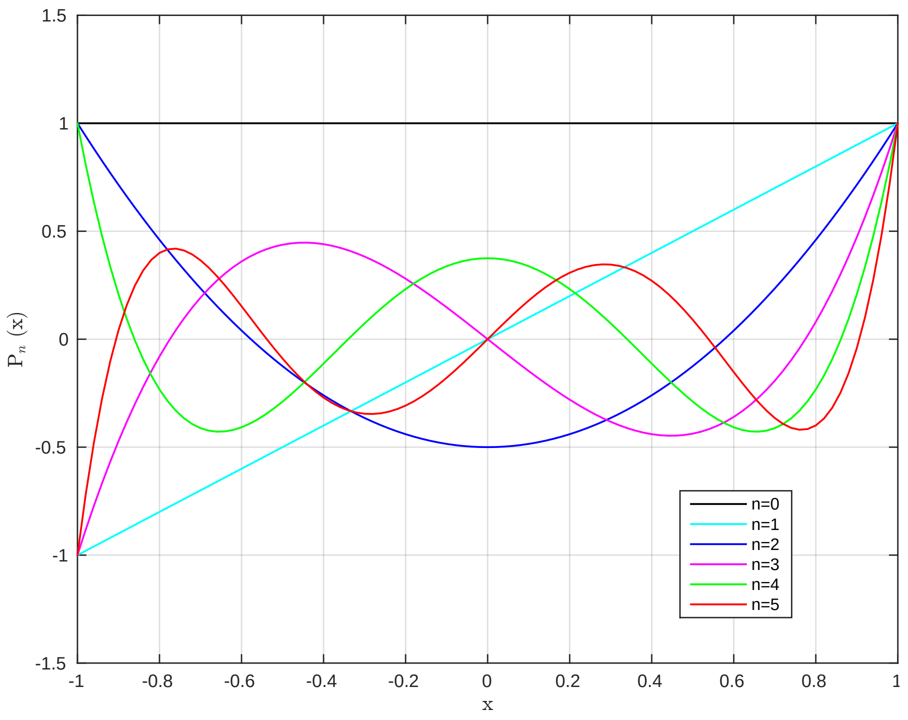


Fig. A.3 Legendre Polynomials

Orthogonality:

Similar with the Jacobi polynomials $P_n^{(\alpha,\beta)}(x)$, Legendre polynomials are orthogonal with respect to the following weight function on the interval $[-1,1]$:

$$w(x) = 1 \tag{A.30}$$

$$\int_{-1}^{1} P_m(x)P_n(x)dx = h_n^2\delta_{mn} = \frac{2}{2n+1}\delta_{mn} \tag{A.31}$$

Appendix B

Derivation of PRKE Coupled with Lumped Parameter TH Model with Generalized PCE

In this Appendix, the stochastic version of the PRKE with lumped parameter TH feedback model is presented using PC method. Recall that we have three random input parameters, which are all expanded with respect to Hermite polynomials in the following form:

$$\rho_{\text{ext}}(\xi_1) = \sum_{i=0}^{P} \rho_{\text{ext}}^{i} \Psi_i(\xi_1) \tag{B.1}$$

$$\alpha_{\text{D}}(\xi_2) = \sum_{i=0}^{P} \alpha_{\text{D}}^{i} \Psi_i(\xi_2) \tag{B.2}$$

$$\alpha_{\text{c}}(\xi_3) = \sum_{i=0}^{P} \alpha_{\text{c}}^{i} \Psi_i(\xi_3) \tag{B.3}$$

where ξ_1 , ξ_2 and ξ_3 are independent and identically distributed (i.i.d.) standard normal random variables. By defining $\boldsymbol{\xi} = [\xi_1, \xi_2, \xi_3]^\intercal$ we have for the four QoIs:

$$p(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} p^{i}(t) \Psi_i(\boldsymbol{\xi}) \tag{B.4}$$

$$C(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} C^{i}(t) \Psi_i(\boldsymbol{\xi}) \tag{B.5}$$

$$T_{\text{fuel}}(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} T_{\text{fuel}}^{i}(t) \Psi_i(\boldsymbol{\xi}) \tag{B.6}$$

$$T_{\text{cool}}(t,\boldsymbol{\xi}) = \sum_{i=0}^{P} T_{\text{cool}}^{i}(t)\Psi_i(\boldsymbol{\xi}) \tag{B.7}$$

Note that the P in QoI expansions can be different from the P in input expansions. Here we use the same symbol for notational conciseness. Substitute the above expansions into the original system of ODEs and the reactivity equation, the stochastic version of the PRKE coupled system is:

$$\frac{dp(t,\boldsymbol{\xi})}{dt} = \frac{\rho(t,\boldsymbol{\xi})-\beta}{\Lambda}p(t,\boldsymbol{\xi}) + \lambda C(t,\boldsymbol{\xi}) \tag{B.8}$$

$$\frac{dC(t,\boldsymbol{\xi})}{dt} = \frac{\beta}{\Lambda}p(t,\boldsymbol{\xi}) - \lambda C(t,\boldsymbol{\xi}) \tag{B.9}$$

$$\frac{dT_{\text{fuel}}(t,\boldsymbol{\xi})}{dt} = \frac{\Omega_{\text{power}}}{\rho_{\text{fuel}}c_{p,\text{fuel}}}p(t,\boldsymbol{\xi}) - \frac{[T_{\text{fuel}}(t,\boldsymbol{\xi}) - T_{\text{cool}}(t,\boldsymbol{\xi})]}{\rho_{\text{fuel}}c_{p,\text{fuel}}\hat{R}_{\text{th}}} \tag{B.10}$$

$$\frac{dT_{\text{cool}}(t,\boldsymbol{\xi})}{dt} = \frac{A_{\text{fuel}}}{A_{\text{cool}}}\frac{[T_{\text{fuel}}(t,\boldsymbol{\xi}) - T_{\text{cool}}(t,\boldsymbol{\xi})]}{\rho_{\text{cool}}c_{p,\text{cool}}\hat{R}_{\text{th}}} - \frac{2u}{H}\left[T_{\text{cool}}(t,\boldsymbol{\xi}) - T_{\text{cool}}^{\text{in}}\right] \tag{B.11}$$

Substituting the expansions into the reactivity equation and simplifying, we obtain:

$$\begin{aligned}
\rho(t,\boldsymbol{\xi}) &= \rho_{\text{ext}}(\xi_1) - \alpha_{\text{D}}(\xi_2)\left[T_{\text{fuel}}(t,\boldsymbol{\xi}) - T_{\text{fuel}}(0)\right] - \alpha_{\text{c}}(\xi_3)\left[T_{\text{cool}}(t,\boldsymbol{\xi}) - T_{\text{cool}}(0)\right] \\
&= \sum_{i=0}^{P}\rho_{\text{ext}}^{i}\Psi_i(\xi_1) - \sum_{i=0}^{P}\alpha_{\text{D}}^{i}\Psi_i(\xi_2)\left[\sum_{l=0}^{P}T_{\text{fuel}}^{l}(t)\Psi_l(\boldsymbol{\xi}) - T_{\text{fuel}}(0)\right] \\
&\qquad - \sum_{i=0}^{P}\alpha_{\text{c}}^{i}\Psi_i(\xi_3)\left[\sum_{l=0}^{P}T_{\text{cool}}^{l}(t)\Psi_l(\boldsymbol{\xi}) - T_{\text{cool}}(0)\right]
\end{aligned} \tag{B.12}$$

With further algebraic substitutions, the power equation becomes:

$$\begin{aligned}
\sum_{i=0}^{P}\frac{dp^i(t)}{dt}\Psi_i(\boldsymbol{\xi}) &= -\frac{\beta}{\Lambda}\sum_{i=0}^{P}p^i(t)\Psi_i(\boldsymbol{\xi}) + \lambda\sum_{i=0}^{P}C^i(t)\Psi_i(\boldsymbol{\xi}) + \frac{1}{\Lambda}\sum_{j=0}^{P}p^j(t)\Psi_j(\boldsymbol{\xi})\cdot \\
&\left[\sum_{i=0}^{P}\rho_{\text{ext}}^{i}\Psi_i(\xi_1) + T_{\text{fuel}}(0)\sum_{i=0}^{P}\alpha_{\text{D}}^{i}\Psi_i(\xi_2) + T_{\text{cool}}(0)\sum_{i=0}^{P}\alpha_{\text{c}}^{i}\Psi_i(\xi_3)\right] - \frac{1}{\Lambda}\sum_{j=0}^{P}p^j(t)\Psi_j(\boldsymbol{\xi})\cdot \\
&\left[\sum_{i=0}^{P}\alpha_{\text{D}}^{i}\Psi_i(\xi_2)\sum_{l=0}^{P}T_{\text{fuel}}^{l}(t)\Psi_l(\boldsymbol{\xi}) + \sum_{i=0}^{P}\alpha_{\text{c}}^{i}\Psi_i(\xi_3)\sum_{l=0}^{P}T_{\text{cool}}^{l}(t)\Psi_l(\boldsymbol{\xi})\right]
\end{aligned} \tag{B.13}$$

The precursor balance equation becomes:

$$\sum_{i=0}^{P} \frac{dC^i(t)}{dt} \Psi_i(\boldsymbol{\xi}) = \frac{\beta}{\Lambda} \sum_{i=0}^{P} p^i(t) \Psi_i(\boldsymbol{\xi}) - \lambda \sum_{i=0}^{P} C^i(t) \Psi_i(\boldsymbol{\xi}) \tag{B.14}$$

The fuel temperature balance equation becomes:

$$\begin{aligned} \sum_{i=0}^{P} \frac{dT^i_{\text{fuel}}(t)}{dt} \Psi_i(\boldsymbol{\xi}) &= \frac{\Omega_{\text{power}}}{\rho_{\text{fuel}} c_{p,\text{fuel}}} \sum_{i=0}^{P} p^i(t) \Psi_i(\boldsymbol{\xi}) \\ &\quad - \frac{1}{\rho_{\text{fuel}} c_{p,\text{fuel}} \hat{R}_{\text{th}}} \left[\sum_{i=0}^{P} T^i_{\text{fuel}}(t) \Psi_i(\boldsymbol{\xi}) - \sum_{i=0}^{P} T^i_{\text{cool}}(t) \Psi_i(\boldsymbol{\xi}) \right] \end{aligned} \tag{B.15}$$

The coolant temperature balance equation becomes:

$$\begin{aligned} \sum_{i=0}^{P} \frac{dT^i_{\text{cool}}(t)}{dt} \Psi_i(\boldsymbol{\xi}) &= \frac{A_{\text{fuel}}}{A_{\text{cool}}} \frac{1}{\rho_{\text{cool}} c_{p,\text{cool}} \hat{R}_{\text{th}}} \left[\sum_{i=0}^{P} T^i_{\text{fuel}}(t) \Psi_i(\boldsymbol{\xi}) - \sum_{i=0}^{P} T^i_{\text{cool}}(t) \Psi_i(\boldsymbol{\xi}) \right] \\ &\quad - \frac{2u}{H} \left[\sum_{i=0}^{P} T^i_{\text{cool}}(t) \Psi_i(\boldsymbol{\xi}) - T^{\text{in}}_{\text{cool}} \right] \end{aligned} \tag{B.16}$$

Finally, the above equations are projected on the s^{th} Hermite polynomial for $s = 0, 1, ..., P$:

$$\begin{aligned} \frac{dp^s(t)}{dt} &= -\frac{\beta}{\Lambda} p^s(t) + \lambda C^s(t) + \frac{1}{\Lambda} \sum_{i=0}^{P} \sum_{j=0}^{P} \mathbf{T}_{ijs} \left[\rho^i_{\text{ext}} + T_{\text{fuel}}(0) \alpha^i_{\text{D}} + T_{\text{cool}}(0) \alpha^i_{\text{c}} \right] p^j(t) \\ &\quad - \frac{1}{\Lambda} \sum_{i=0}^{P} \sum_{j=0}^{P} \sum_{l=0}^{P} \mathbf{T}_{ijls} \left[\alpha^i_{\text{D}} T^l_{\text{fuel}}(t) + \alpha^i_{\text{c}} T^l_{\text{cool}}(t) \right] p^j(t) \end{aligned} \tag{B.17}$$

$$\frac{dC^s(t)}{dt} = \frac{\beta}{\Lambda} p^s(t) - \lambda C^s(t) \tag{B.18}$$

$$\frac{dT^s_{\text{fuel}}(t)}{dt} = \frac{\Omega_{\text{power}}}{\rho_{\text{fuel}} c_{p,\text{fuel}}} p^s(t) - \frac{1}{\rho_{\text{fuel}} c_{p,\text{fuel}} \hat{R}_{\text{th}}} \left[T^s_{\text{fuel}}(t) - T^s_{\text{cool}}(t) \right] \tag{B.19}$$

$$\frac{dT^s_{\text{cool}}(t)}{dt} = \frac{A_{\text{fuel}}}{A_{\text{cool}}} \frac{1}{\rho_{\text{cool}} c_{p,\text{cool}} \hat{R}_{\text{th}}} \left[T^s_{\text{fuel}}(t) - T^s_{\text{cool}}(t) \right] - \frac{2u}{H} \left[T^s_{\text{cool}}(t) - T^{\text{in}}_{\text{cool}} \cdot \delta_{0s} \right] \tag{B.20}$$

Note that in the power equation, we have two tensor terms $\mathbf{T}_{ijs}$ and $\mathbf{T}_{ijls}$. They are due to the strong non-linearity of the model. If we have the following double summation:

$$a(\xi)b(\xi) = \sum_{i=0}^{P}\sum_{j=0}^{P} a_i b_j \Psi_i(\xi)\Psi_j(\xi) = \sum_{s=0}^{P} f_s \Psi_s(\xi) \tag{B.21}$$

Projecting it on the s^{th} Hermite polynomial corresponds to performing the following operation:

$$f_s = \sum_{i=0}^{P}\sum_{j=0}^{P} a_i b_j \frac{\langle \Psi_i, \Psi_j, \Psi_s \rangle}{\langle \Psi_s, \Psi_s \rangle} = \sum_{i=0}^{P}\sum_{j=0}^{P} a_i b_j \mathbf{T}_{ijs} \tag{B.22}$$

Therefore the three-dimensional tensor $\mathbf{T}_{ijs}$ is defined as:

$$\mathbf{T}_{ijs} = \frac{\langle \Psi_i, \Psi_j, \Psi_s \rangle}{\langle \Psi_s, \Psi_s \rangle} \tag{B.23}$$

Similarly, for triple summation we need a four dimension tensor:

$$a(\xi)b(\xi)c(\xi) = \sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{l=0}^{P} a_i b_j c_l \Psi_i(\xi)\Psi_j(\xi)\Psi_l(\xi) = \sum_{s=0}^{P} f_s \Psi_s(\xi) \tag{B.24}$$

Projecting it on the s^{th} Hermite polynomial:

$$f_s = \sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{l=0}^{P} a_i b_j c_l \frac{\langle \Psi_i, \Psi_j, \Psi_l, \Psi_s \rangle}{\langle \Psi_s, \Psi_s \rangle} = \sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{l=0}^{P} a_i b_j c_l \mathbf{T}_{ijls} \tag{B.25}$$

The four-dimensional tensor $\mathbf{T}_{ijs}$ is defined as:

$$\mathbf{T}_{ijls} = \frac{\langle \Psi_i, \Psi_j, \Psi_l, \Psi_s \rangle}{\langle \Psi_s, \Psi_s \rangle} \tag{B.26}$$

Appendix C

List of TRACE Physical Model Parameters

In this appendix, we provide the full list for 36 physical model parameters implemented in TRACE that can be calibrated. These parameters are referred to as UQ Sensitivity Coefficients in TRACE manual [138]. All of them are **Multiplicative** factors, with only one exception that parameter `filmTransBoilTMin` can be **Scalar**, **Additive** or **Multiplicative**. The nominal values for all the **Multiplicative** factors are 1.0. HTC stands for heat transfer coefficient.

Table C.1 List of 36 physical model parameters implemented in TRACE

ID	Mnemonic Name	Description
1000	`bubSlugLiqIntHTC`	Liquid to interface bubbly-slug HTC
1001	`annMistLiqIntHTC`	Liquid to interface annular-mist HTC
1002	`transLiqIntHTC`	Liquid to interface transition HTC
1003	`stratLiqIntHTC`	Liquid to interface stratified HTC
1004	`bubSlugVapIntHTC`	Vapor to interface bubbly-slug HTC
1005	`annMistVapIntHTC`	Vapor to interface annular-mist HTC
1006	`transVapIntHTC`	Vapor to interface transition HTC
1007	`stratVapIntHTC`	Vapor to interface stratified HTC
1008	`singlePhaseLiqWallHTC`	Single phase liquid to wall HTC
1009	`singlePhaseVapWallHTC`	Single phase vapor to wall HTC
1010	`filmTransBoilTMin`	Film to transition boiling T_{min} criterion temperature
1011	`dispFlowFilmBoilHTC`	Dispersed flow film boiling HTC
1012	`subBoilHTC`	Subcooled boiling HTC
1013	`nucBoilHTC`	Nucleate boiling HTC

Table C.1 List of 36 physical model parameters implemented in TRACE

ID	Mnemonic Name	Description
1014	`DNB_CHF`	Departure from nucleate boiling / critical heat flux
1015	`transBoilHTC`	Transition boiling heat transfer coefficient
1016	`gapConductance`	Gap conductance coefficient
1017	`fuelThermalCond`	Fuel thermal conductivity
1018	`cladMWRX`	Cladding metal-water reaction rate coefficient
1019	`fuelRodIntPress`	Rod internal pressure coefficient
1020	`burstTemp`	Burst temperature coefficient
1021	`burstStrain`	Burst strain coefficient
1022	`wallDrag`	Wall drag coefficient
1023	`formLoss`	Form loss coefficient
1024	`bubblyIntDrag`	Interfacial drag (bubbly) coefficient
1027	`dropletIntDrag`	Interfacial drag (droplet) coefficient
1028	`bubSlugIntDragBundle`	Interfacial drag (bubbly/slug Rod Bundle - Bestion) coefficient
1029	`bubSlugIntDragVessel`	Interfacial drag (bubbly/slug Vessel) coefficient
1030	`annMistIntDragVessel`	Interfacial drag (annular/mist Vessel) coefficient
1031	`dffbIntDrag`	Interfacial drag (dispersed flow film boiling) coefficient
1032	`invSlugIntDrag`	Interfacial drag (inverted slug flow) coefficient
1033	`invAnnIntDrag`	Interfacial drag (inverted annular flow) coefficient
1034	`tempFlood`	Flooding coefficient temperature coefficient
1035	`lengthFlood`	Flooding coefficient length coefficient
1036	`invAnnVapWallHTC`	Vapor to wall inverted annular HTC
1037	`invAnnLiqWallHTC`	Liquid to wall inverted annular HTC

Appendix D

Linear Least Squares

In this appendix, some theory for linear least squares regression will be presented, which is highly relevant for Gaussian Process modeling. The simple linear regression will be introduced first, followed by multiple linear regression and Generalized Least Squares.

Define Y which is a scalar as dependent variable, also called response. Also define a set of independent variables $\{X_1, X_2, \ldots, X_d\}$, also called regressors or predictors. We would like to quantify the relationship between $\{X_1, X_2, \ldots, X_d\}$ and the mean of Y in the form of a function, which can be used to predict Y.

D.1 Simple Linear Regression

For simple linear regression, we start with only one predictor X and one response Y. The mean function $\mathbb{E}(Y|X)$ expresses the expected value of Y as a function of X. Regression models focus on estimation of this mean function from data. There is also a variance function $\mathrm{Var}(Y|X)$ expresses the variance of Y as a function of X. As we will treat X as a fixed and known value rather than a random variable, we can write the mean and variance function as $\mathbb{E}(Y)$ and $\mathrm{Var}(Y)$ for notational convenience.

The simple linear regression model assumes the mean function as:

$$\mathbb{E}(Y) = \beta_0 + \beta_1 X \tag{D.1}$$

where β_0 is the intercept and β_1 is the slope. Together they are called the coefficients of the linear regression.

The variance function is assumed to be a constant that does not depend on X:

$$\mathrm{Var}(Y) = \sigma^2 \tag{D.2}$$

Given n observed data points that come in pairs:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \tag{D.3}$$

In theory, we regard the y_i values as realizations of random variables Y. For purposes of regression, we regard the x_i values as known constants. The model equation for simple linear regression is:

$$y_i = \beta_0 + \beta_1 x_i + e_i, \quad i = 1, 2, \ldots, n \tag{D.4}$$

where e_i is the displacement of y_i from its mean.

The Gauss-Markov conditions require $\{e_1, e_2, \ldots, e_n\}$ to:

- have mean zero:
$$\mathbb{E}(e_i) = 0$$
- be homoscedastic:
$$\mathrm{Var}(e_i) = \sigma^2$$
- be uncorrelated:
$$\mathrm{Cov}(e_i, e_j) = 0 \quad \text{for} \quad i \neq j$$

Next, we need to evaluate the parameters β_0 and β_1 based on data. Denote $\widehat{\beta_0}$ and $\widehat{\beta_1}$ as their estimates respectively. Also denote the i^{th} fitted value at x_i (also called predicted value at x_i, or the prediction of y_i) as:

$$\widehat{y_i} = \widehat{\beta_0} + \widehat{\beta_1} x_i \tag{D.5}$$

The difference between y_i and its prediction $\widehat{y_i}$ is called the i^{th} residual:

$$\widehat{e_i} = y_i - \widehat{y_i} \tag{D.6}$$

The least squares estimate of β_0 and β_1 in simple linear regression are the values that minimize the Residual Sum of Squares (RSS):

$$\mathrm{RSS}\left(\beta_0, \beta_1\right) = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \beta_1 x_i\right)^2 \tag{D.7}$$

One method of finding the minimizer is to differentiate with respect to β_0 and β_1 , set the derivatives equal to 0, and solve

$$\frac{\partial \mathrm{RSS}\left(\beta_0, \beta_1\right)}{\partial \beta_0} = -2 \sum_{i=1}^{n} \left(y_i - \beta_0 - \beta_1 x_i\right) = 0 \tag{D.8}$$

$$\frac{\partial \text{RSS}(\beta_0,\beta_1)}{\partial \beta_1} = -2\sum_{i=1}^{n} x_i (y_i - \beta_0 - \beta_1 x_i) = 0 \tag{D.9}$$

Re-arrange terms, we get:

$$\beta_0 n + \beta_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \tag{D.10}$$

$$\beta_0 \sum_{i=1}^{n} x_i + \beta_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i \tag{D.11}$$

Solve for β_0 and β_1:

$$\widehat{\beta_0} = \bar{y} - \widehat{\beta_1}\bar{x} \tag{D.12}$$

$$\widehat{\beta_1} = \frac{\sum x_i y_i - n\bar{x}\bar{y}}{\sum x_i^2 - n\bar{x}^2} \tag{D.13}$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ adn $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. Using the following notations:

$$\text{SXX} = \sum_{i=1}^{n} (x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2 \tag{D.14}$$

$$\text{SXY} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) = \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y} \tag{D.15}$$

We have the estimate $\widehat{\beta_0}$ and $\widehat{\beta_1}$ as:

$$\widehat{\beta_0} = \bar{y} - \widehat{\beta_1}\bar{x} \tag{D.16}$$

$$\widehat{\beta_1} = \frac{\text{SXY}}{\text{SXX}} \tag{D.17}$$

D.2 Multiple Linear Regression

Multiple linear regression model consider d independent variables $\{X_1, X_2, \ldots, X_d\}$ in the model for response Y. The mean function takes the form:

$$\mathbb{E}(Y) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_d X_d \tag{D.18}$$

The variance function:

$$\text{Var}(Y) = \sigma^2 \tag{D.19}$$

The n observations come in with the form of:

$$\begin{gathered}(y_1, x_{11}, x_{12}, x_{13}, \ldots, x_{1d}) \\ (y_2, x_{21}, x_{22}, x_{23}, \ldots, x_{2d}) \\ \ldots \\ (y_n, x_{n1}, x_{n2}, x_{n3}, \ldots, x_{nd})\end{gathered}$$

The scalar model equation for multiple linear regression is:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_d x_{id} + e_i, \quad i = 1, 2, \ldots, n \tag{D.20}$$

where $\{e_1, e_2, \ldots, e_n\}$ still satisfy the Gauss-Markov conditions.

Define the following notations:

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \quad \mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_d \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1d} \\ 1 & x_{21} & x_{22} & \ldots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \ldots & x_{nd} \end{bmatrix}$$

Then the regression equations can be written together in matrix form:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e} \tag{D.21}$$

Similarly for the mean and variance functions:

$$\mathbb{E}(\mathbf{Y}) = \mathbf{X}\boldsymbol{\beta}, \qquad \mathrm{Var}(Y) = \sigma^2 \mathbf{I} \tag{D.22}$$

Again we would like to find the estimator $\widehat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ given the data:

$$\widehat{\boldsymbol{\beta}} = \left[\widehat{\beta_0},\ \widehat{\beta_1},\ \widehat{\beta_2},\ \ldots,\ \widehat{\beta_d}\right]^\top \tag{D.23}$$

Then the vector of fitted values (or predicted values) and residuals are:

$$\widehat{\mathbf{y}} = \mathbf{X}\widehat{\boldsymbol{\beta}} \tag{D.24}$$

$$\widehat{\mathbf{e}} = \mathbf{y} - \widehat{\mathbf{y}} \tag{D.25}$$

where $\mathbf{y}$ is a realization of $\mathbf{Y}$, in this case, the set of observed responses $\mathbf{y} = [y_1, y_2, \ldots, y_d]^\top$.

The RSS function is:

$$\begin{aligned}\mathrm{RSS}(\boldsymbol{\beta}) &= \mathbf{e}^\top\mathbf{e} = (\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) \\ &= \mathbf{y}^\top\mathbf{y}+\boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta}-2\mathbf{y}^\top\mathbf{X}\boldsymbol{\beta}\end{aligned} \tag{D.26}$$

To minimize the RSS function, differentiate it with respect to $\boldsymbol{\beta}$ and set the derivative to $\mathbf{0}$, we get the normal equations:

$$\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^\top\mathbf{y} \tag{D.27}$$

Note that the normal equations can be direct result of requiring the residual vector $\mathbf{e}$ to be orthogonal to every column of $\mathbf{X}$. If the inverse of $\mathbf{X}^\top\mathbf{X}$ exists, as it will if the columns of $\mathbf{X}$ are linearly independent, the least squares estimates are unique and are given by:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y} \tag{D.28}$$

The estimator $\widehat{\boldsymbol{\beta}}$ is the Best Linear Unbiased Estimator (BLUE) for $\boldsymbol{\beta}$. Using the rules for means and variances of random vectors, we can find the mean and variance of this estimator:

$$\begin{aligned}\mathbb{E}\left(\widehat{\boldsymbol{\beta}}\right) &= \mathbb{E}\left(\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{Y}|\mathbf{X}\right) \\ &= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbb{E}(\mathbf{Y}|\mathbf{X}) \\ &= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}\end{aligned} \tag{D.29}$$

which shows that $\widehat{\boldsymbol{\beta}}$ is an unbiased estimator of $\boldsymbol{\beta}$.

$$\begin{aligned}\mathrm{Var}\left(\widehat{\boldsymbol{\beta}}\right) &= \mathrm{Var}\left(\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{Y}|\mathbf{X}\right) \\ &= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathrm{Var}(\mathbf{Y}|\mathbf{X})\,\mathbf{X}\left(\mathbf{X}^\top\mathbf{X}\right)^{-1} \\ &= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\left[\sigma^2\mathbf{I}\right]\mathbf{X}\left(\mathbf{X}^\top\mathbf{X}\right)^{-1} \\ &= \sigma^2\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\end{aligned} \tag{D.30}$$

D.3 Generalized Least Squares for Linear Regression

In previous sections for simple and multiple linear regressions, we have assumed that e_i's have equal variances and are uncorrelated $\mathrm{Var}(\mathbf{e}) = \sigma^2\mathbf{I}$. Now we consider the case where this condition is not satisfied. It still assumed that $\mathbb{E}(\mathbf{e}) = \mathbf{0}$, but the variance:

$$\mathrm{Var}(\mathbf{e}) = \sigma^2 \boldsymbol{\Sigma} \tag{D.31}$$

for some invertible matrix $\boldsymbol{\Sigma}$ (symmetric and positive definite) and some constant σ^2.

It is obvious that unless $\boldsymbol{\Sigma}$ is a multiple of the identity matrix $\mathbf{I}$, the Gauss-Markov conditions are not satisfied, and the ordinary least squares estimator shown in Equation is not necessarily a BLUE.

To derive the BLUE, the generalized least squares method is used given matrix $\boldsymbol{\Sigma}$. Instead of minimizing RSS, we minimize the Generalized Residual Sum of Squares (GRSS) function:

$$\mathrm{GRSS}(\boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \tag{D.32}$$

The corresponding GLS estimate for $\boldsymbol{\beta}$ that minimize GRSS is:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^\top \boldsymbol{\Sigma}^{-1} \mathbf{X}\right)^{-1} \mathbf{X}^\top \boldsymbol{\Sigma}^{-1} \mathbf{y} \tag{D.33}$$

The corresponding GLS estimator is a BLUE for this generalized linear regression. To prove this, consider a matrix decomposition of $\boldsymbol{\Sigma}^{-1}$:

$$\boldsymbol{\Sigma}^{-1} = \mathbf{C}^\top \mathbf{C} \tag{D.34}$$

$\mathbf{C}$ is a square matrix and:

$$\mathbf{C}^{-1} \left(\mathbf{C}^\top\right)^{-1} = \boldsymbol{\Sigma} \tag{D.35}$$

Multiply both sides of Equation D.21 by $\mathbf{C}$:

$$\mathbf{CY} = \mathbf{CX}\boldsymbol{\beta} + \mathbf{Ce} \tag{D.36}$$

Define:

$$\mathbf{Z} = \mathbf{CY}, \quad \mathbf{M} = \mathbf{CX}, \quad \boldsymbol{\delta} = \mathbf{Ce} \tag{D.37}$$

The model is transformed to:

$$\mathbf{Z} = \mathbf{M}\boldsymbol{\beta} + \boldsymbol{\delta} \tag{D.38}$$

For $\boldsymbol{\delta}$ we have:

$$\mathbb{E}(\boldsymbol{\delta}) = \mathbb{E}(\mathbf{Ce}) = \mathbf{C}\mathbb{E}(\mathbf{e}) = \mathbf{0} \tag{D.39}$$

$$\mathrm{Var}(\boldsymbol{\delta}) = \mathrm{Var}(\mathbf{Ce}) = \mathbf{C}\mathrm{Var}(\mathbf{e})\mathbf{C}^\top = \mathbf{C}\sigma^2\boldsymbol{\Sigma}\mathbf{C}^\top = \sigma^2\mathbf{I} \tag{D.40}$$

Apparently, for the transformed model, the Gauss-Markov conditions are satisfied. The BLUE (Equation D.33) is then obtained as the transformed model's least squares estimator:

$$\begin{aligned}\widehat{\boldsymbol{\beta}} &= \left(\mathbf{M}^\top\mathbf{M}\right)^{-1}\mathbf{M}^\top\mathbf{z} \\ &= \left(\mathbf{X}^\top\mathbf{C}^\top\mathbf{C}\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{C}^\top\mathbf{C}\mathbf{y} \\ &= \left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{y}\end{aligned} \tag{D.41}$$

where $\mathbf{z} = \mathbf{C}\mathbf{y}$. Moreover, this estimator minimizes:

$$\begin{aligned}(\mathbf{z}-\mathbf{M}\boldsymbol{\beta})^\top(\mathbf{z}-\mathbf{M}\boldsymbol{\beta}) &= [\mathbf{C}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})]^\top[\mathbf{C}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})] \\ &= (\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^\top\mathbf{C}^\top\mathbf{C}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) \\ &= (\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) \\ &= \mathrm{GRSS}(\boldsymbol{\beta})\end{aligned} \tag{D.42}$$

The mean of this estimator:

$$\begin{aligned}\mathbb{E}\left(\widehat{\boldsymbol{\beta}}\right) &= \mathbb{E}\left(\left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{Y}|\mathbf{X}\right) \\ &= \left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbb{E}(\mathbf{Y}|\mathbf{X}) \\ &= \left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}\end{aligned} \tag{D.43}$$

which shows that $\widehat{\boldsymbol{\beta}}$ is an unbiased estimator of $\boldsymbol{\beta}$.

The variance of this estimator:

$$\begin{aligned}\mathrm{Var}\left(\widehat{\boldsymbol{\beta}}\right) &= \mathrm{Var}\left(\left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{Y}|\mathbf{X}\right) \\ &= \left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathrm{Var}(\mathbf{Y}|\mathbf{X})\boldsymbol{\Sigma}^{-1}\mathbf{X}\left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1} \\ &= \left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\left[\sigma^2\boldsymbol{\Sigma}\right]\boldsymbol{\Sigma}^{-1}\mathbf{X}\left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1} \\ &= \sigma^2\left(\mathbf{X}^\top\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\end{aligned} \tag{D.44}$$

Appendix E

Derivation of the Inverse of Block Matrix

In this appendix, some general formula for matrix inversion in block form are presented.

1. Matrix Inversion Lemma:
   Let $\mathbf{W}$ be a non-singular $n \times n$ matrix, $\mathbf{Y}$ be a non-singular $m \times m$ matrix, $\mathbf{X}$ and $\mathbf{Z}$ be arbitrary $n \times m$ and $m \times n$ matrix such that $\left(\mathbf{Y}^{-1} + \mathbf{Z}\mathbf{W}^{-1}\mathbf{X}\right)$ is a non-singular $m \times m$ matrix. then

$$(\mathbf{W} + \mathbf{XYZ})^{-1} = \mathbf{W}^{-1} - \mathbf{W}^{-1}\mathbf{X}\left(\mathbf{Y}^{-1} + \mathbf{Z}\mathbf{W}^{-1}\mathbf{X}\right)^{-1}\mathbf{Z}\mathbf{W}^{-1} \tag{E.1}$$

2. Matrix Inversion in Block form:
   Let a $(m+n) \times (m+n)$ matrix $\mathbf{M}$ be partitioned into a block form:

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_{m\times m} & \mathbf{B}_{m\times n} \\ \mathbf{C}_{n\times m} & \mathbf{D}_{n\times n} \end{bmatrix} \tag{E.2}$$

   Suppose $\mathbf{M}$ is non-singular and it has inverse matrix:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{E}_{m\times m} & \mathbf{F}_{m\times n} \\ \mathbf{G}_{n\times m} & \mathbf{H}_{n\times n} \end{bmatrix} \tag{E.3}$$

   It will be shown how to derive elements of $\mathbf{M}^{-1}$ in terms of elements of $\mathbf{M}$. One sufficient condition for the non-singularity of $\mathbf{M}$ is that matrix $\mathbf{A}$ and matrix $\mathbf{C}$ are invertible. However, in general they are not necessary conditions.

Given invertible matrices **A** and matrix **C**:

$$\begin{aligned}\mathbf{M}\mathbf{M}^{-1} &= \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{bmatrix}\begin{bmatrix}\mathbf{E} & \mathbf{F}\\ \mathbf{G} & \mathbf{H}\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{AE}+\mathbf{BG} & \mathbf{AF}+\mathbf{BH}\\ \mathbf{CE}+\mathbf{DG} & \mathbf{CF}+\mathbf{DH}\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{I}_m & \mathbf{0}_{m\times n}\\ \mathbf{0}_{n\times m} & \mathbf{I}_n\end{bmatrix}\end{aligned} \tag{E.4}$$

where **I** denotes the unit matrix and **0** a zero matrix,

Now we have the following system of equations:

$$\mathbf{AE}+\mathbf{BG}=\mathbf{I}_m \tag{E.5}$$
$$\mathbf{AF}+\mathbf{BH}=\mathbf{0}_{m\times n} \tag{E.6}$$
$$\mathbf{CE}+\mathbf{DG}=\mathbf{0}_{n\times m} \tag{E.7}$$
$$\mathbf{CF}+\mathbf{DH}=\mathbf{I}_n \tag{E.8}$$

It follows from Equations E.6 and E.7 that:

$$\mathbf{F}=-\mathbf{A}^{-1}\mathbf{BH} \tag{E.9}$$
$$\mathbf{G}=-\mathbf{D}^{-1}\mathbf{CE} \tag{E.10}$$

Substitute Equations E.9 and E.10 into Equations E.5 and E.8

$$\left(\mathbf{A}-\mathbf{BD}^{-1}\mathbf{C}\right)\mathbf{E}=-\mathbf{I}_m \tag{E.11}$$
$$\left(\mathbf{D}-\mathbf{CA}^{-1}\mathbf{B}\right)\mathbf{H}=-\mathbf{I}_n \tag{E.12}$$

We get:

$$\mathbf{E}=\left(\mathbf{A}-\mathbf{BD}^{-1}\mathbf{C}\right)^{-1} \tag{E.13}$$
$$\mathbf{H}=\left(\mathbf{D}-\mathbf{CA}^{-1}\mathbf{B}\right)^{-1} \tag{E.14}$$

Substitute Equations E.13 and E.14 into Equations E.9 and E.10:

$$\mathbf{F}=-\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D}-\mathbf{CA}^{-1}\mathbf{B}\right)^{-1} \tag{E.15}$$

$$\mathbf{G} = -\mathbf{D}^{-1}\mathbf{C}\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} \tag{E.16}$$

Finally, we have:

$$\begin{aligned}\mathbf{M}^{-1} &= \begin{bmatrix}\mathbf{E} & \mathbf{F}\\ \mathbf{G} & \mathbf{H}\end{bmatrix}\\ &= \begin{bmatrix}\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & -\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\\ -\mathbf{D}^{-1}\mathbf{C}\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & \left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\end{bmatrix}\end{aligned} \tag{E.17}$$

Also, since $\mathbf{M}\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{M}$, we can also get:

$$\mathbf{M}^{-1} = \begin{bmatrix}\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & -\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1}\mathbf{B}\mathbf{D}^{-1}\\ -\left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\mathbf{C}\mathbf{A}^{-1} & \left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\end{bmatrix} \tag{E.18}$$

It can be proved that Equation E.17 and Equation E.18 are equivalent.

3. In the above it is mentioned that invertible matrices **A** and **C** are sufficient for the non-singularity of **M** but not necessary. Now consider the case when **A** is not invertible. $\mathbf{A}^{-1}$ does not exist so we need to modify the equation for $\mathbf{M}^{-1}$ as shown in Equation E.17. Based on Equation E.1:

$$\begin{aligned}\left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1} &= \mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{C}\left(\mathbf{B}\mathbf{D}^{-1}\mathbf{C} - \mathbf{A}\right)^{-1}\mathbf{B}\mathbf{D}^{-1}\\ &= \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\end{aligned} \tag{E.19}$$

where $\mathbf{Q} = \left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)$, which is non-singular. Replacing $\left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}$ with Equation E.19 in Equation E.17:

$$\mathbf{M}^{-1} = \begin{bmatrix}\mathbf{Q}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\right)\\ -\mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1} & \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\end{bmatrix} \tag{E.20}$$

Also, from Equation E.19:

$$\begin{aligned}\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B} &= \left(\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\right)^{-1}\\ \mathbf{A}^{-1}\mathbf{B} &= \mathbf{C}^{-1}\left[\mathbf{D} - \left(\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\right)^{-1}\right]\end{aligned} \tag{E.21}$$

Substituting into Equation E.20, we get:

$$\begin{aligned}-\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D}^{-1}+\mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\right) &= -\mathbf{C}^{-1}\mathbf{D}\left(\mathbf{D}^{-1}+\mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\right)+\mathbf{C}^{-1} \\ &= -\mathbf{C}^{-1}\mathbf{D}\mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1} \\ &= -\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\end{aligned} \tag{E.22}$$

Finally, we have:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{Q}^{-1} & -\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1} \\ -\mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1} & \mathbf{D}^{-1}+\mathbf{D}^{-1}\mathbf{C}\mathbf{Q}^{-1}\mathbf{B}\mathbf{D}^{-1}\end{bmatrix} \tag{E.23}$$

4. Given that matrix $\mathbf{A}$ is not invertible, we further assume that $\mathbf{A}=\mathbf{0}$:

$$\mathbf{Q} = \mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C} = -\mathbf{B}\mathbf{D}^{-1}\mathbf{C} \tag{E.24}$$

And we require that $\mathbf{Q}$ is a non-singular matrix. Equation E.23 becomes:

$$\begin{aligned}\mathbf{M}^{-1} &= \begin{bmatrix}\mathbf{0} & \mathbf{B} \\ \mathbf{C} & \mathbf{D}\end{bmatrix}^{-1} \\ &= \begin{bmatrix} -\left(\mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & \left(\mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1}\mathbf{B}\mathbf{D}^{-1} \\ \mathbf{D}^{-1}\mathbf{C}\left(\mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & \mathbf{D}^{-1}-\mathbf{D}^{-1}\mathbf{C}\left(\mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1}\mathbf{B}\mathbf{D}^{-1}\end{bmatrix}\end{aligned} \tag{E.25}$$

For the application in Universal Kriging, we will have $\mathbf{B}=\mathbf{C}^\top$. In this case the inverse for block matrix is:

$$\begin{aligned}\mathbf{M}^{-1} &= \begin{bmatrix}\mathbf{0} & \mathbf{C}^\top \\ \mathbf{C} & \mathbf{D}\end{bmatrix}^{-1} \\ &= \begin{bmatrix} -\left(\mathbf{C}^\top\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & \left(\mathbf{C}^\top\mathbf{D}^{-1}\mathbf{C}\right)^{-1}\mathbf{C}^\top\mathbf{D}^{-1} \\ \mathbf{D}^{-1}\mathbf{C}\left(\mathbf{C}^\top\mathbf{D}^{-1}\mathbf{C}\right)^{-1} & \mathbf{D}^{-1}-\mathbf{D}^{-1}\mathbf{C}\left(\mathbf{C}^\top\mathbf{D}^{-1}\mathbf{C}\right)^{-1}\mathbf{C}^\top\mathbf{D}^{-1}\end{bmatrix}\end{aligned} \tag{E.26}$$

Appendix F

Derivation of the GP Predictor and MSE

In this appendix, the prediction formula of the GP metamodel at an unknown point $\mathbf{x}^*$ is derived, as well as the MSE of this prediction. They are denoted by $\hat{y}(\mathbf{x}^*)$ and $\text{MSE}\left[\hat{y}(\mathbf{x}^*)\right]$ respectively. The notations $\mu_{\hat{y}}(\mathbf{x}^*)$ and $\sigma^2_{\hat{y}}(\mathbf{x}^*)$ are also widely used to emphasis that GP metamodel at untried input $\mathbf{x}^*$ follows a Gaussian distribution, and the GP predictor and MSE are the mean value and variance respectively. In this thesis, we use these two sets of notations interchangeably and assume no difference between them.

F.1 Derivation of the Predictor

The UK metamodel takes the form of:

$$y(\mathbf{x}) = \sum_{j=1}^{n} \beta_j f_j(\mathbf{x}) + z(\mathbf{x}) = \mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta} + z(\mathbf{x}) \tag{F.1}$$

where $\mathbf{f}(\mathbf{x})$ is the set of known regression/basis functions evaluated at a general input point $\mathbf{x}$, and $\boldsymbol{\beta}$ is the vector of unnknown regression coefficients as shown in Table 4.2. The definitions of $\mathbf{F}$, $\mathbf{R}$ and $\mathbf{r}(\mathbf{x}^*)$ (to appear) are also same with those defined in Table 4.2.

Given the design sites $\mathbf{X}$ and the corresponding output values $\mathbf{y}$, our goal is to find the best linear unbiased predictor (BLUP) $\hat{y}(\mathbf{x}^*)$ at an untried input $\mathbf{x}^*$. We start with the general linear regression predictor $\hat{y}(\mathbf{x}^*) = \mathbf{c}^\top\mathbf{y}$ and solve for the one that has minimal MSE.

$$\begin{aligned}\hat{y}(\mathbf{x}^*) - y(\mathbf{x}^*) &= \mathbf{c}^\top\mathbf{y} - y(\mathbf{x}^*) \\ &= \mathbf{c}^\top(\mathbf{F}\boldsymbol{\beta} + \mathbf{z}) - \left(\mathbf{f}^\top(\mathbf{x}^*)\boldsymbol{\beta} + z(\mathbf{x}^*)\right) \\ &= \mathbf{c}^\top\mathbf{z} - z(\mathbf{x}^*) + \left(\mathbf{F}^\top\mathbf{c} - \mathbf{f}(\mathbf{x}^*)\right)^\top\boldsymbol{\beta}\end{aligned} \tag{F.2}$$

where $\mathbf{z} = \left[z(\mathbf{x}^{(1)}), z(\mathbf{x}^{(2)}), \ldots, z(\mathbf{x}^{(m)})\right]^\top$ is the vector of the residuals of this linear predictor at the design sites.

To enforce the unbiasedness of this linear predictor, we require:

$$\mathbb{E}\left[\hat{y}(\mathbf{x}^*)\right] = \mathbb{E}\left[y(\mathbf{x}^*)\right] \tag{F.3}$$

$$\mathbf{c}^\top \mathbf{F}\boldsymbol{\beta} = \mathbf{f}^\top(\mathbf{x}^*)\boldsymbol{\beta} \tag{F.4}$$

$$\mathbf{F}^\top \mathbf{c} - \mathbf{f}(\mathbf{x}^*) = \mathbf{0} \tag{F.5}$$

Equation F.5 holds because the vector $\boldsymbol{\beta}$ cannot be a zero vector. The MSE for this linear predictor is defined as:

$$\begin{aligned}
\text{MSE}\left[\hat{y}(\mathbf{x}^*)\right] &= \mathbb{E}\left[\left(\hat{y}(\mathbf{x}^*) - y(\mathbf{x}^*)\right)^2\right] \\
&= \mathbb{E}\left[\left(\mathbf{c}^\top \mathbf{z} - z(\mathbf{x}^*) + \left(\mathbf{F}^\top \mathbf{c} - \mathbf{f}(\mathbf{x}^*)\right)^\top \boldsymbol{\beta}\right)^2\right] \\
&= \mathbb{E}\left[\left(\mathbf{c}^\top \mathbf{z} - z(\mathbf{x}^*)\right)^2\right] \\
&= \mathbb{E}\left[\mathbf{c}^\top \mathbf{z}\mathbf{z}^\top \mathbf{c} - 2\mathbf{c}^\top \mathbf{z} z(\mathbf{x}^*) + z^2(\mathbf{x}^*)\right] \\
&= \mathbf{c}^\top \mathbb{E}\left[\mathbf{z}\mathbf{z}^\top\right] \mathbf{c} - 2\mathbf{c}^\top \mathbb{E}\left[\mathbf{z} z(\mathbf{x}^*)\right] + \mathbb{E}\left[z^2(\mathbf{x}^*)\right]
\end{aligned} \tag{F.6}$$

Since $z(\mathbf{x})$ is a zero-mean stationary Gaussian process with covariance:

$$\text{Cov}\left[z(\mathbf{x}^{(i)}), z(\mathbf{x}^{(j)})\right] = \sigma^2 \mathcal{R}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)$$

we have:

$$\mathbb{E}\left[\mathbf{z}\mathbf{z}^\top\right] = \sigma^2 \mathbf{R} \tag{F.7}$$

$$\mathbb{E}\left[\mathbf{z} z(\mathbf{x}^*)\right] = \sigma^2 \mathbf{r}(\mathbf{x}^*) \tag{F.8}$$

$$\mathbb{E}\left[z^2(\mathbf{x}^*)\right] = \sigma^2 \tag{F.9}$$

Therefore, Equation F.6 becomes:

$$\text{MSE}\left[\hat{y}(\mathbf{x}^*)\right] = \sigma^2\left[1 + \mathbf{c}^\top \mathbf{R}\mathbf{c} - 2\mathbf{c}^\top \mathbf{r}(\mathbf{x}^*)\right] \tag{F.10}$$

To minimize the MSE shown in Equation F.10 with respect to **c** under the constraint of unbiasedness denoted by Equation F.5, we define the Lagrangian function:

$$L(\mathbf{c},\boldsymbol{\lambda}) = \sigma^2\left[1+\mathbf{c}^\top\mathbf{R}\mathbf{c}-2\mathbf{c}^\top\mathbf{r}(\mathbf{x}^*)\right]-\boldsymbol{\lambda}^\top\left(\mathbf{F}^\top\mathbf{c}-\mathbf{f}(\mathbf{x}^*)\right) \tag{F.11}$$

The gradient (Jacobian matrix) of Equation F.11 with respect to **c** is:

$$\boldsymbol{J}_{\mathbf{c}}(L) = 2\sigma^2\left(\mathbf{R}\mathbf{c}-\mathbf{r}(\mathbf{x}^*)\right)-\mathbf{F}\boldsymbol{\lambda} \tag{F.12}$$

Requiring it to be **0** we get:

$$\mathbf{R}\mathbf{c}-\mathbf{r}(\mathbf{x}^*) = \frac{\mathbf{F}\boldsymbol{\lambda}}{2\sigma^2} \tag{F.13}$$

The system of equations to be solved can be written in the following form:

$$\begin{bmatrix}\mathbf{0} & \mathbf{F}^\top\\ \mathbf{F} & \mathbf{R}\end{bmatrix}\begin{bmatrix}-\frac{\boldsymbol{\lambda}}{2\sigma^2}\\ \mathbf{c}\end{bmatrix} = \begin{bmatrix}\mathbf{f}(\mathbf{x}^*)\\ \mathbf{r}(\mathbf{x}^*)\end{bmatrix} \tag{F.14}$$

The solution to Equation F.14 is:

$$\begin{bmatrix}-\frac{\boldsymbol{\lambda}}{2\sigma^2}\\ \mathbf{c}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{F}^\top\\ \mathbf{F} & \mathbf{R}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{f}(\mathbf{x}^*)\\ \mathbf{r}(\mathbf{x}^*)\end{bmatrix} \tag{F.15}$$

Using Equation E.26 from Appendix E:

$$\begin{bmatrix}\mathbf{0} & \mathbf{F}^\top\\ \mathbf{F} & \mathbf{R}\end{bmatrix}^{-1} = \begin{bmatrix}-\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1} & \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\\ \mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1} & \mathbf{R}^{-1}-\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\end{bmatrix} \tag{F.16}$$

The solution for **c** is:

$$\begin{aligned}\mathbf{c} &= \mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{f}(\mathbf{x}^*)+\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)-\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)\\ &= \mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)-\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)-\mathbf{f}(\mathbf{x}^*)\right)\end{aligned} \tag{F.17}$$

The linear predictor, or the BLUP becomes:

$$\hat{y}(\mathbf{x}^*) = \mathbf{c}^\top\mathbf{y} = \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{y}-\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*)-\mathbf{f}(\mathbf{x}^*)\right)^\top\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{y} \tag{F.18}$$

Note that we have used the matrix symmetry properties:

$$\left(\mathbf{R}^{-1}\right)^\top = \mathbf{R}^{-1}$$

$$\left(\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\right)^\top = \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}$$

Based on the theory on least squares estimate in Appendix D (Equation D.33) we know that the Best Linear Unbiased Estimator (BLUE) for $\boldsymbol{\beta}$ is:

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{y} \tag{F.19}$$

It follows that:

$$\begin{aligned}\hat{y}(\mathbf{x}^*) &= \mu_{\hat{y}}(\mathbf{x}^*)\\ &= \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{y} - \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right)^\top\hat{\boldsymbol{\beta}}\\ &= \mathbf{f}^\top(\mathbf{x}^*)\hat{\boldsymbol{\beta}} + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)\end{aligned} \tag{F.20}$$

Equation F.20 is the prediction formula for UK metamodel.

F.2 Derivation of the MSE

Based on Equations F.5, F.10, F.13 and F.15:

$$\begin{aligned}\mathrm{MSE}\left[\hat{y}(\mathbf{x}^*)\right] &= \sigma^2_{\hat{y}}(\mathbf{x}^*)\\ &= \sigma^2\left[1 + \mathbf{c}^\top\mathbf{R}\mathbf{c} - 2\mathbf{c}^\top\mathbf{r}(\mathbf{x}^*)\right]\\ &= \sigma^2\left[1 + \mathbf{c}^\top\left(\mathbf{r}(\mathbf{x}^*) + \mathbf{F}\frac{\boldsymbol{\lambda}}{2\sigma^2}\right) - 2\mathbf{c}^\top\mathbf{r}(\mathbf{x}^*)\right]\\ &= \sigma^2\left[1 + \mathbf{c}^\top\mathbf{r}(\mathbf{x}^*) + \mathbf{f}^\top(\mathbf{x}^*)\frac{\boldsymbol{\lambda}}{2\sigma^2} - 2\mathbf{c}^\top\mathbf{r}(\mathbf{x}^*)\right]\\ &= \sigma^2\left[1 + \mathbf{f}^\top(\mathbf{x}^*)\frac{\boldsymbol{\lambda}}{2\sigma^2} - \mathbf{c}^\top\mathbf{r}(\mathbf{x}^*)\right]\\ &= \sigma^2\left[1 + \mathbf{f}^\top(\mathbf{x}^*)\frac{\boldsymbol{\lambda}}{2\sigma^2} - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{c}\right]\\ &= \sigma^2\left[1 - \begin{bmatrix}\mathbf{f}^\top(\mathbf{x}^*) & \mathbf{r}^\top(\mathbf{x}^*)\end{bmatrix}\begin{bmatrix}-\frac{\boldsymbol{\lambda}}{2\sigma^2}\\ \mathbf{c}\end{bmatrix}\right]\\ &= \sigma^2\left[1 - \begin{bmatrix}\mathbf{f}^\top(\mathbf{x}^*) & \mathbf{r}^\top(\mathbf{x}^*)\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{F}^\top\\ \mathbf{F} & \mathbf{R}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{f}(\mathbf{x}^*)\\ \mathbf{r}(\mathbf{x}^*)\end{bmatrix}\right]\end{aligned} \tag{F.21}$$

Equation F.21 is the MSE or variance of the prediction $\hat{y}(\mathbf{x}^*)$. This is the form of MSE used in [119] and [121].

To get the expanded form of the MSE instead of the matrix version as shown in Equation F.21, we expand the matrix:

$$
\begin{aligned}
&\begin{bmatrix} \mathbf{f}^\top(\mathbf{x}^*) & \mathbf{r}^\top(\mathbf{x}^*) \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{F}^\top \\ \mathbf{F} & \mathbf{R} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}(\mathbf{x}^*) \\ \mathbf{r}(\mathbf{x}^*) \end{bmatrix} = \\
&\begin{bmatrix} -\mathbf{f}^\top(\mathbf{x}^*)\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1} + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1} \\ \mathbf{f}^\top(\mathbf{x}^*)\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1} + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1} - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1} \end{bmatrix}^\top \begin{bmatrix} \mathbf{f}(\mathbf{x}^*) \\ \mathbf{r}(\mathbf{x}^*) \end{bmatrix} \\
&= -\,\mathbf{f}^\top(\mathbf{x}^*)\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{f}(\mathbf{x}^*) + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{f}(\mathbf{x}^*) + \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) \\
&\quad + \mathbf{f}^\top(\mathbf{x}^*)\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) \\
&= \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) + \mathbf{f}^\top(\mathbf{x}^*)\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right) \\
&\quad - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{F}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right) \\
&= \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) + \left(\mathbf{f}^\top(\mathbf{x}^*) - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{F}\right)\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right)
\end{aligned}
$$

Note that both the first and second part of the above equation are scalars.

$$
\begin{aligned}
&\begin{bmatrix} \mathbf{f}^\top(\mathbf{x}^*) & \mathbf{r}^\top(\mathbf{x}^*) \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{F}^\top \\ \mathbf{F} & \mathbf{R} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}(\mathbf{x}^*) \\ \mathbf{r}(\mathbf{x}^*) \end{bmatrix} = \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \\
&\qquad \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right)^\top \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1} \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right)
\end{aligned}
\tag{F.22}
$$

Finally, the MSE becomes:

$$
\begin{aligned}
\mathrm{MSE}\left[\hat{y}(\mathbf{x}^*)\right] = \sigma_{\hat{y}}^2(\mathbf{x}^*) = \sigma^2 \Big[1 - \mathbf{r}^\top(\mathbf{x}^*)\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) + \\
\left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right)^\top \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\right)^{-1} \left(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) - \mathbf{f}(\mathbf{x}^*)\right) \Big]
\end{aligned}
\tag{F.23}
$$

Equation F.23 is a convenient form of the MSE that has also been widely used.

References

[1] Adams, B. M., Ebeida, M. S., Eldred, M. S., Geraci, G., Jakeman, J. D., Maupin, K. A., and et al. (2016a). Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.5 user's manual. Technical report, Sandia National Laboratories, Tech. Rep. SAND2014-4633, Updated November 11, 2016.

[2] Adams, B. M., Ebeida, M. S., Eldred, M. S., Geraci, G., Jakeman, J. D., Maupin, K. A., and et al (2016b). Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.5 theory manual. Technical report, Sandia National Laboratories, Tech. Rep. SAND2014-4253, Updated November 11, 2016.

[3] Andrieu, C. and Thoms, J. (2008). A tutorial on adaptive mcmc. *Statistics and Computing*, 18(4):343–373.

[4] Arendt, P. D., Apley, D. W., and Chen, W. (2012). Quantification of model uncertainty: Calibration, model discrepancy, and identifiability. *Journal of Mechanical Design*, 134(10):100908.

[5] Babuška, I., Nobile, F., and Tempone, R. (2007). A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034.

[6] Bachoc, F. (2013). Cross validation and maximum likelihood estimations of hyper-parameters of gaussian processes with model misspecification. *Computational Statistics & Data Analysis*, 66:55–69.

[7] Bachoc, F., Bois, G., Garnier, J., and Martinez, J.-M. (2014). Calibration and improved prediction of computer models by universal kriging. *Nuclear Science and Engineering*, 176(1):81–97.

[8] Barthelmann, V., Novak, E., and Ritter, K. (2000). High dimensional polynomial interpolation on sparse grids. *Advances in Computational Mathematics*, 12(4):273–288.

[9] Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for gaussian process emulators. *Technometrics*, 51(4):425–438.

[10] Bayarri, M. J., Berger, J. O., Paulo, R., Sacks, J., Cafeo, J. A., Cavendish, J., Lin, C.-H., and Tu, J. (2007). A framework for validation of computer models. *Technometrics*, 49(2):138–154.

[11] Bestion, D. (1990). The physical closure laws in the cathare code. *Nuclear Engineering and Design*, 124(3):229–245.

[12] Bratley, P., Fox, B. L., and Niederreiter, H. (1992). Implementation and tests of low-discrepancy sequences. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 2(3):195–213.

[13] Brynjarsdóttir, J. and O'Hagan, A. (2014). Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, 30(11):114007.

[14] Bui, A., Dinh, N., Nourgaliev, R., and Youngblood, R. (2013). Two-phase flow and heat transfer model calibration and code validation-a subcooled boiling flow case study. *submitted to NURETH*, 15.

[15] Bui, A., Williams, B., and Dinh, N. (2014). Advanced calibration and validation of a mechanistic model of subcooled boiling two-phase flow. In *Proceedings of ICAPP-2014*. Charlotte, USA, April 6-9, 2014.

[16] Burwell, M., Lerchl, G., Miro, J., Teschendorff, V., and Wolfert, K. (1989). The thermal-hydraulic code athlet for analysis of pwr and bwr systems. In *Fourth international topical meeting on nuclear reactor thermal-hydraulics (NURETH-4). Proceedings. Vol. 2*.

[17] Cacuci, D. G. and Arslan, E. (2014). Reducing uncertainties via predictive modeling: Flica4 calibration using bfbt benchmarks. *Nuclear Science and Engineering*, 176(3):339–349.

[18] Cameron, R. H. and Martin, W. T. (1947). The orthogonal development of non-linear functionals in series of fourier-hermite functionals. *Annals of Mathematics*, pages 385–392.

[19] Campbell, K. (2006). Statistical calibration of computer simulations. *Reliability Engineering & System Safety*, 91(10):1358–1363.

[20] Chen, W., Xiong, Y., Tsui, K.-L., and Wang, S. (2008). A design-driven validation approach using bayesian prediction models. *Journal of Mechanical Design*, 130(2):021101.

[21] Chkifa, M. A. (2013). On the lebesgue constant of leja sequences for the complex unit disk and of their real projection. *Journal of Approximation Theory*, 166:176–200.

[22] Conrad, P. R. and Marzouk, Y. M. (2013). Adaptive smolyak pseudospectral approximations. *SIAM Journal on Scientific Computing*, 35(6):A2643–A2670.

[23] Constantine, P. G., Eldred, M. S., and Phipps, E. T. (2012). Sparse pseudospectral approximation method. *Computer Methods in Applied Mechanics and Engineering*, 229:1–12.

[24] Cressie, N. (1990). The origins of kriging. *Mathematical geology*, 22(3):239–252.

[25] Cressie, N. (2015). *Statistics for spatial data*. John Wiley & Sons, revised edition.

[26] Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1991). Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86(416):953–963.

[27] D'Auria, F., Camargo, C., and Mazzantini, O. (2012). The best estimate plus uncertainty (bepu) approach in licensing of current nuclear reactors. *Nuclear Engineering and Design*, 248:317–328.

[28] de Crécy, A. (2001). Determination of the uncertainties of the constitutive relationships of the cathare 2 code. In *Proceedings of M&C-2001*.

[29] Debusschere, B. J., Najm, H. N., Pébay, P. P., Knio, O. M., Ghanem, R. G., and Le Maître, O. P. (2004). Numerical challenges in the use of polynomial chaos representations for stochastic processes. *SIAM journal on scientific computing*, 26(2):698–719.

[30] Eldred, M. and Burkardt, J. (2009). Comparison of non-intrusive polynomial chaos and stochastic collocation methods for uncertainty quantification. *AIAA paper*, 976(2009):1–20.

[31] Eldred, M. S. (2009). Recent advances in non-intrusive polynomial chaos and stochastic collocation methods for uncertainty analysis and design. *AIAA Paper*, 2274(2009):37.

[32] Eldred, M. S., Webster, C. G., and Constantine, P. (2008). Evaluation of non-intrusive approaches for wiener-askey generalized polynomial chaos. In *Proceedings of the 10th AIAA Non-Deterministic Approaches Conference, number AIAA-2008-1892, Schaumburg, IL*, volume 117, page 189.

[33] Emonot, P., Souyri, A., Gandrille, J., and Barré, F. (2011). Cathare-3: A new system code for thermal-hydraulics in the context of the neptune project. *Nuclear Engineering and Design*, 241(11):4476–4481.

[34] Ernst, O. G., Mugler, A., Starkloff, H.-J., and Ullmann, E. (2012). On the convergence of generalized polynomial chaos expansions. *ESAIM: Mathematical Modelling and Numerical Analysis*, 46(2):317–339.

[35] Evensen, G. (2009). *Data assimilation: the ensemble Kalman filter*. Springer Science & Business Media.

[36] Fletcher, C. and Schultz, R. (1995). Relap5/mod3 code manual volume v: User's guidelines. *Idaho National Engineering Laboratory, Lockheed Idaho Technologies Company, Idaho Falls, Idaho*, 83415.

[37] Forrester, A. I., Bressloff, N. W., and Keane, A. J. (2006). Optimization using surrogate models and partially converged computational fluid dynamics simulations. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 462, pages 2177–2204. The Royal Society.

[38] Forrester, A. I. and Keane, A. J. (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79.

[39] Frangos, M., Marzouk, Y., Willcox, K., and van Bloemen Waanders, B. (2010). *Surrogate and reduced-order modeling: a comparison of approaches for large-scale statistical inverse problems*. Chapter 7, Large-Scale Inverse Problems and Quantification of Uncertainty, John Wiley & Sons.

[40] Ganapathysubramanian, B. and Zabaras, N. (2007). Sparse grid collocation schemes for stochastic natural convection problems. *Journal of Computational Physics*, 225(1):652–685.

[41] Gaston, D., Newman, C., Hansen, G., and Lebrun-Grandie, D. (2009). Moose: A parallel computational framework for coupled systems of nonlinear equations. *Nuclear Engineering and Design*, 239(10):1768–1778.

[42] Gattiker, J. R. (2008). Gaussian process models for simulation analysis (gpm/sa) command, function, and data structure reference. *Technical Report LA-UR-08-08057, Los Alamos National Laboratory, Los Alamos, New Mexico.*

[43] Geelhood, K., Luscher, W. G., and Beyer, C. (2011). *FRAPCON-3.4: A Computer Code for the Calculation of Steady State Thermal-mechanical Behavior of Oxide Fuel Rods for*

High Burnup. US Nuclear Regulatory Commission, Office of Nuclear Regulatory Research Richland, WA.

[44] Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2014). *Bayesian data analysis*. Chapman & Hall/CRC Boca Raton, FL, USA, third edition.

[45] Gelman, A., Roberts, G., and Gilks, W. (1996). Efficient metropolis jumping hules. *Bayesian statistics*, 5(599-608):42.

[46] Gerstner, T. and Griebel, M. (1998). Numerical integration using sparse grids. *Numerical algorithms*, 18(3-4):209–232.

[47] Gerstner, T. and Griebel, M. (2003). Dimension–adaptive tensor–product quadrature. *Computing*, 71(1):65–87.

[48] Ghanem, R. (1999). Ingredients for a general purpose stochastic finite elements implementation. *Computer Methods in Applied Mechanics and Engineering*, 168(1):19–34.

[49] Ghanem, R. G. and Spanos, P. D. (2003). *Stochastic finite elements: a spectral approach*. Courier Corporation.

[50] Gilks, W. R., Richardson, S., and Spiegelhalter, D. (1995). *Markov chain Monte Carlo in practice*. CRC press.

[51] Gilli, L., Lathouwers, D., Kloosterman, J. L., and van der Hagen, T. (2012). Performing uncertainty analysis of a nonlinear point-kinetics/lumped parameters problem using polynomial chaos techniques. *Annals of Nuclear Energy*, 40(1):35–44.

[52] Glen, G. and Isaacs, K. (2012). Estimating sobol sensitivity indices using correlations. *Environmental Modelling & Software*, 37:157–166.

[53] Glück, M. (2008). Validation of the sub-channel code f-cobra-tf: Part ii. recalculation of void measurements. *Nuclear Engineering and Design*, 238(9):2317–2327.

[54] Guba, A., Makai, M., and Pál, L. (2003). Statistical aspects of best estimate method—i. *Reliability engineering & system safety*, 80(3):217–232.

[55] Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive metropolis algorithm. *Bernoulli*, pages 223–242.

[56] Hales, J., Novascone, S., Spencer, B., Williamson, R., Pastore, G., and Perez, D. (2014). Verification of the bison fuel performance code. *Annals of Nuclear Energy*, 71:81–90.

[57] Han, G., Santner, T. J., and Rawlinson, J. J. (2009). Simultaneous determination of tuning and calibration parameters for computer experiments. *Technometrics*, 51(4):464–474.

[58] Helton, J. C. and Davis, F. (2002). Illustration of sampling-based methods for uncertainty and sensitivity analysis. *Risk Analysis*, 22(3):591–622.

[59] Helton, J. C. and Davis, F. J. (2003). Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System Safety*, 81(1):23–69.

[60] Helton, J. C., Johnson, J. D., Sallaberry, C. J., and Storlie, C. B. (2006). Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering & System Safety*, 91(10):1175–1209.

[61] Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008). Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103(482):570–583.

[62] Higdon, D., Geelhood, K., Williams, B., and Unal, C. (2013). Calibration of tuning parameters in the frapcon model. *Annals of Nuclear Energy*, 52:95–102.

[63] Higdon, D., Kennedy, M., Cavendish, J. C., Cafeo, J. A., and Ryne, R. D. (2004). Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing*, 26(2):448–466.

[64] Housiadas, C. (2002). Lumped parameters analysis of coupled kinetics and thermal-hydraulics for small reactors. *Annals of Nuclear Energy*, 29(11):1315–1325.

[65] Hu, G. and Kozlowski, T. (2016). Inverse uncertainty quantification of trace physical model parameters using bfbt benchmark data. *Annals of Nuclear Energy*, 96:197–203.

[66] Iooss, B., Boussouf, L., Feuillard, V., and Marrel, A. (2010). Numerical studies of the metamodel fitting and validation processes. *International Journal of Advances in Systems and Measurements*, 3:11–21.

[67] Jaeger, W., Sánchez Espinoza, V. H., Montero Mayorga, F. J., and Queral, C. (2013). Uncertainty and sensitivity studies with trace-susa and trace-dakota by means of steady state bfbt data. *Science and Technology of Nuclear Installations*, 2013.

[68] Johnson, M. E., Moore, L. M., and Ylvisaker, D. (1990). Minimax and maximin distance designs. *Journal of statistical planning and inference*, 26(2):131–148.

[69] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

[70] Joseph, V. R., Hung, Y., and Sudjianto, A. (2008). Blind kriging: A new method for developing metamodels. *Journal of mechanical design*, 130(3):031102.

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

[72] Kilgour, W., Turnbull, J., White, R., Bull, A., Jackson, P., and Palmer, I. (1991). Capabilities and validation of the enigma fuel performance code. *ANS Avignon, April*.

[73] Killeen, J., Turnbull, J., and Sartori, E. (2006). Fuel modelling at extended burnup: Iaea coordinated research project fumex-ii. In *Transactions of the Top Fuel 2006 International Meeting on LWR Fuel Performance*, pages 22–26.

[74] Kleijnen, J. P. (2009). Kriging metamodeling in simulation: a review. *European Journal of Operational Research*, 192(3):707–716.

[75] Klimke, A. and Wohlmuth, B. (2005). Algorithm 847: spinterp: Piecewise multilinear hierarchical sparse grid interpolation in matlab. *ACM Transactions on Mathematical Software (TOMS)*, 31(4):561–579.

[76] Lassmann, K., Schubert, A., and van de Laar, J. (2000). Transuranus handbook. *Document Number Version*, 1.

[77] Lataniotis, C., Marelli, S., and Sudret, B. (2015). Uqlab user manual–kriging (gaussian process modelling). *Report UQLab-V0*, pages 9–105.

[78] Le Maître, O. and Knio, O. M. (2010). *Spectral methods for uncertainty quantification: with applications to computational fluid dynamics*. Springer Science & Business Media.

[79] Ling, Y., Mullins, J., and Mahadevan, S. (2014). Selection of model discrepancy priors in bayesian calibration. *Journal of Computational Physics*, 276:665–680.

[80] Liu, F., Bayarri, M., Berger, J., et al. (2009). Modularization in bayesian analysis, with emphasis on analysis of computer models. *Bayesian Analysis*, 4(1):119–150.

[81] Lophaven, S. N., Nielsen, H. B., and Søndergaard, J. (2002). Dace-a matlab kriging toolbox, version 2.0. Technical report.

[82] Ma, X. and Zabaras, N. (2009). An efficient bayesian inference approach to inverse problems based on an adaptive sparse grid collocation method. *Inverse Problems*, 25(3):035013.

[83] Marelli, S. and Sudret, B. (2014). Uqlab: a framework for uncertainty quantification in matlab. In *Vulnerability, Uncertainty, and Risk: Quantification, Mitigation, and Management*, pages 2554–2563.

[84] Marrel, A., Iooss, B., Laurent, B., and Roustant, O. (2009). Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety*, 94(3):742–751.

[85] Martin, J. D. (2009). Computational improvements to estimating kriging metamodel parameters. *Journal of Mechanical Design*, 131(8):084501.

[86] Martin, J. D. and Simpson, T. W. (2005). Use of kriging models to approximate deterministic computer models. *AIAA journal*, 43(4):853–863.

[87] Marzouk, Y. and Xiu, D. (2009). A stochastic collocation approach to bayesian inference in inverse problems. *Communications in Computational Physics*, 6(4):826–847.

[88] Marzouk, Y. M. and Najm, H. N. (2009). Dimensionality reduction and polynomial chaos acceleration of bayesian inference in inverse problems. *Journal of Computational Physics*, 228(6):1862–1902.

[89] Marzouk, Y. M., Najm, H. N., and Rahn, L. A. (2007). Stochastic spectral methods for efficient bayesian solution of inverse problems. *Journal of Computational Physics*, 224(2):560–586.

[90] Matheron, G. (1963). Principles of geostatistics. *Economic geology*, 58(8):1246–1266.

[91] McFarland, J., Mahadevan, S., Romero, V., and Swiler, L. (2008). Calibration and uncertainty analysis for computer simulations with multivariate output. *AIAA journal*, 46(5):1253–1265.

[92] McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61.

[93] Morokoff, W. J. and Caflisch, R. E. (1994). Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15(6):1251–1279.

[94] Morris, M. D. and Mitchell, T. J. (1995). Exploratory designs for computational experiments. *Journal of statistical planning and inference*, 43(3):381–402.

[95] Morrison, R. E., Bryant, C. M., Terejanu, G., Prudhomme, S., and Miki, K. (2013). Data partition methodology for validation of predictive models. *Computers & Mathematics with Applications*, 66(10):2114–2125.

[96] Mullins, J., Mahadevan, S., and Urbina, A. (2016). Optimal test selection for prediction uncertainty reduction. *Journal of Verification, Validation and Uncertainty Quantification*, 1(4):041002.

[97] Najm, H. N. (2009). Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 41:35–52.

[98] Neykov, B., Aydogan, F., Hochreiter, L., Ivanov, K., Utsuno, H., Kasahara, F., Sartori, E., and Martin, M. (2005). *NUPEC BWR full-size fine-mesh bundle test (BFBT) benchmark*. OECD/NEA, NEA/NSC/DOC(2005)5.

[99] Nguyen, T. N. and Downar, T. J. (2017). Surrogate-based multi-experiment calibration of the bison fission gas behavior model. *Nuclear Engineering and Design*, 320:409–417.

[100] Niederreiter, H. (1992). *Random number generation and quasi-Monte Carlo methods*. SIAM.

[101] Nobile, F., Tempone, R., and Webster, C. G. (2008). A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345.

[102] Oberkampf, W. L. and Roy, C. J. (2010). *Verification and validation in scientific computing*. Cambridge University Press.

[103] Oberkampf, W. L. and Trucano, T. G. (2002). Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences*, 38(3):209–272.

[104] O'Hagan, A. (2006). Bayesian analysis of computer code outputs: a tutorial. *Reliability Engineering & System Safety*, 91(10):1290–1300.

[105] Pastore, G., Hales, J., Novascone, S., Perez, D., Spencer, B., and Williamson, R. (2013a). Analysis of fission gas release in lwr fuel using the bison code. In *Proc. of the LWR Fuel Performance Meeting*.

[106] Pastore, G., Luzzi, L., Di Marcello, V., and Van Uffelen, P. (2013b). Physics-based modelling of fission gas swelling and release in uo2 applied to integral fuel rod analysis. *Nuclear Engineering and Design*, 256:75–86.

[107] Pastore, G., Swiler, L., Hales, J. D., Novascone, S. R., Perez, D. M., Spencer, B. W., Luzzi, L., Van Uffelen, P., and Williamson, R. L. (2015). Uncertainty and sensitivity analysis of fission gas behavior in engineering-scale fuel modeling. *Journal of Nuclear Materials*, 456:398–408.

[108] Perez, D. M., Williamson, R., Novascone, S., Pastore, G., Hales, J., and Spencer, B. (2013). Assessment of bison: A nuclear fuel performance analysis code. Technical report, Tech. Rep. INL/MIS-13-30314, Idaho National Laboratory.

[109] Petruzzi, A. and D'Auria, F. (2008). Thermal-hydraulic system codes in nulcear reactor safety and qualification procedures. *Science and Technology of Nuclear Installations*, 2008.

[110] Qian, P. Z. and Wu, C. J. (2008). Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments. *Technometrics*, 50(2):192–204.

[111] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P. K. (2005). Surrogate-based analysis and optimization. *Progress in aerospace sciences*, 41(1):1–28.

[112] Ragusa, J. C. and Mahadevan, V. S. (2009). Consistent and accurate schemes for coupled neutronics thermal-hydraulics reactor analysis. *Nuclear Engineering and Design*, 239(3):566–579.

[113] Rashid, Y., Dunham, R., and Montgomery, R. (2004). Fuel analysis and licensing code: Falcon mod01. *EPRI Report*, 1011308.

[114] Rasmussen, C. E. and Nickisch, H. (2010). Gaussian processes for machine learning (gpml) toolbox. *Journal of Machine Learning Research*, 11(Nov):3011–3015.

[115] Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian processes for machine learning*, volume 1. MIT press Cambridge.

[116] Roth, G. A. and Aydogan, F. (2014a). Theory and implementation of nuclear safety system codes–part i: Conservation equations, flow regimes, numerics and significant assumptions. *Progress in Nuclear Energy*, 76:160–182.

[117] Roth, G. A. and Aydogan, F. (2014b). Theory and implementation of nuclear safety system codes–part ii: System code closure relations, validation, and limitations. *Progress in Nuclear Energy*, 76:55–72.

[118] Roustant, O., Ginsbourger, D., and Deville, Y. (2012). Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodelling and optimization. *Journal of Statistical Software*, 51(1):54p.

[119] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical science*, pages 409–423.

[120] Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (2008). *Global sensitivity analysis: the primer*. John Wiley & Sons.

[121] Santner, T. J., Williams, B. J., and Notz, W. I. (2003). *The design and analysis of computer experiments*. Springer Science & Business Media.

[122] Sartori, E., Killeen, J., and Turnbull, J. A. (2010). *International Fuel Performance Experiments (IFPE) Database*. Available at: The Organisation for Economic Co-operation and Development (OECD) - Nuclear Energy Agency (NEA), https://www.oecd-nea.org/science/wprs/fuel/ifpelst.html.

[123] Schobi, R., Sudret, B., and Wiart, J. (2015). Polynomial-chaos-based kriging. *International Journal for Uncertainty Quantification*, 5(2).

[124] Shlens, J. (2014). A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*.

[125] Shrestha, R. and Kozlowski, T. (2016). Inverse uncertainty quantification of input model parameters for thermal-hydraulics simulations using expectation–maximization under bayesian framework. *Journal of Applied Statistics*, 43(6):1011–1026.

[126] Smith, R. C. (2013). *Uncertainty quantification: theory, implementation, and applications*, volume 12. Siam.

[127] Smolyak, S. A. (1963). Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Dokl. Akad. Nauk SSSR*, volume 4, page 123.

[128] Sobol, I. M. (1976). Uniformly distributed sequences with an additional uniform property. *USSR Computational Mathematics and Mathematical Physics*, 16(5):236–242.

[129] Ştefănescu, R., Schmidt, K., Hite, J., Smith, R. C., and Mattingly, J. (2017). Hybrid optimization and bayesian inference techniques for a non-smooth radiation detection problem. *International Journal for Numerical Methods in Engineering*.

[130] Stein, M. L. (2012). *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

[131] Stoyanov, M. (2015). User manual: Tasmanian sparse grids. Technical report, Technical report ORNL/TM-2015/596, Oak Ridge National Laboratory, Computer Science and Mathematics Division, Oak Ridge, TN.

[132] Stoyanov, M. K. and Webster, C. G. (2016). A dynamically adaptive sparse grids method for quasi-optimal interpolation of multidimensional functions. *Computers & Mathematics with Applications*, 71(11):2449–2465.

[133] Stripling, H., McClarren, R., Kuranz, C., Grosskopf, M., Rutter, E., and Torralva, B. (2013). A calibration and data assimilation method using the bayesian mars emulator. *Annals of Nuclear Energy*, 52:103–112.

[134] Stuart, A. M. (2010). Inverse problems: a bayesian perspective. *Acta Numerica*, 19:451–559.

[135] Trefethen, L. N. (2008). Is gauss quadrature better than clenshaw-curtis? *SIAM review*, 50(1):67–87.

[136] Trucano, T. G., Swiler, L. P., Igusa, T., Oberkampf, W. L., and Pilch, M. (2006). Calibration, validation, and sensitivity analysis: What's what. *Reliability Engineering & System Safety*, 91(10):1331–1357.

[137] Unal, C., Williams, B., Hemez, F., Atamturktur, S., and McClure, P. (2011). Improved best estimate plus uncertainty methodology, including advanced validation concepts, to license evolving nuclear reactors. *Nuclear Engineering and Design*, 241(5):1813–1833.

[138] USNRC (2014). *TRAC/RELAP Advanced Computational Engine (TRACE) V5.840 User's Manual, Volume 1: Input Specification*. Division of Safety Analysis, Office of Nuclear Regulatory Research, U. S. Nuclear Regulatory Commission, Washington, DC.

[139] Van Oijen, M., Rougier, J., and Smith, R. (2005). Bayesian calibration of process-based forest models: bridging the gap between models and data. *Tree Physiology*, 25(7):915–927.

[140] Wang, J. and Zabaras, N. (2004). A bayesian inference approach to the inverse heat conduction problem. *International Journal of Heat and Mass Transfer*, 47(17):3927–3941.

[141] Wang, S., Chen, W., and Tsui, K.-L. (2009). Bayesian validation of computer models. *Technometrics*, 51(4):439–451.

[142] Wicaksono, D. C., Zerkak, O., and Pautz, A. (2016). Bayesian calibration of thermal-hydraulics model with time-dependent output. In *11th International Topical Meeting on Nuclear Reactor Thermal Hydraulics, Operation and Safety (NUTHOS-11)*, number EPFL-CONF-222443.

[143] Wiener, N. (1938). The homogeneous chaos. *American Journal of Mathematics*, 60(4):897–936.

[144] Wilkinson, R. D. (2010). Bayesian calibration of expensive multivariate computer experiments. *Large-Scale Inverse Problems and Quantification of Uncertainty*, pages 195–215.

[145] Williams, B. and Gattiker, J. (2006). Using the gaussian process model for simulation analysis (gpm/sa) code. Technical report.

[146] Williamson, R., Gamble, K., Perez, D., Novascone, S., Pastore, G., Gardner, R., Hales, J., Liu, W., and Mai, A. (2016). Validating the bison fuel performance code to integral lwr experiments. *Nuclear Engineering and Design*, 301:232–244.

[147] Williamson, R., Hales, J., Novascone, S., Tonks, M., Gaston, D., Permann, C., Andrs, D., and Martineau, R. (2012). Multidimensional multiphysics simulation of nuclear fuel behavior. *Journal of Nuclear Materials*, 423(1):149–163.

[148] Wilson, G. E. (2013). Historical insights in the development of best estimate plus uncertainty safety analysis. *Annals of Nuclear Energy*, 52:2–9.

[149] Wu, X. and Kozlowski, T. (2016). Inverse uncertainty quantification of reactor simulation with polynomial chaos surrogate model. In *Transactions of American Nuclear Society*. American Nuclear Society, New Orleans, LA, USA, June 12-16, 2016.

[150] Wu, X. and Kozlowski, T. (2017a). Inverse uncertainty quantification of reactor simulations under the bayesian framework using surrogate models constructed by polynomial chaos expansion. *Nuclear Engineering and Design*, 313:29–52.

[151] Wu, X. and Kozlowski, T. (2017b). Inverse uncertainty quantification of trace physical model parameters with model discrepancy. In *Transactions of American Nuclear Society*. American Nuclear Society, Washington, DC, USA, October 29 - November 2.

[152] Wu, X. and Kozlowski, T. (2017c). Investigation of adaptive markov chain monte carlo algorithms for inverse uncertainty quantification. In *Proceedings of M&C-2017*. Jeju, Korea, April 16-20, 2017.

[153] Wu, X. and Kozlowski, T. (2017d). Kriging-based inverse uncertainty quantification of bison fission gas release model. In *Transactions of American Nuclear Society*. American Nuclear Society, San Francisco, CA, USA, June 11-15.

[154] Wu, X., Kozlowski, T., and Hales, J. D. (2015). Neutronics and fuel performance evaluation of accident tolerant fecral cladding under normal operation conditions. *Annals of Nuclear Energy*, 85:763–775.

[155] Wu, X., Kozlowski, T., and Meidani, H. (2018). Kriging-based inverse uncertainty quantification of nuclear fuel performance code bison fission gas release model using time series measurement data. *Reliability Engineering & System Safety*, 169:422–436.

[156] Wu, X., Mui, T., Hu, G., Meidani, H., and Kozlowski, T. (2017a). Inverse uncertainty quantification of trace physical model parameters using sparse gird stochastic collocation surrogate model. *Nuclear Engineering and Design*, 319:185–200.

[157] Wu, X., Sabharwall, P., Hales, J., and Kozlowski, T. (2014). Neutronics and fuel performance evaluation of accident tolerant fuel under normal operation conditions. Technical report, Technical Report INL/EXT-14-32591, Idaho National Laboratory, Idaho Falls, ID, USA.

[158] Wu, X., Wang, C., and Kozlowski, T. (2017b). Global sensitivity analysis of trace physical model parameters based on bfbt benchmark. In *Proceedings of M&C-2017*. Jeju, Korea, April 16-20, 2017.

[159] Wu, X., Wang, C., and Kozlowski, T. (2017c). Kriging-based surrogate model for uncertainty quantification and sensitivity analysis. In *Proceedings of M&C-2017*. Jeju, Korea, April 16-20, 2017.

[160] Xiu, D. (2007). Efficient collocational approach for parametric uncertainty analysis. *Commun. Comput. Phys*, 2(2):293–309.

[161] Xiu, D. (2010). *Numerical methods for stochastic computations: a spectral method approach*. Princeton University Press.

[162] Xiu, D. and Hesthaven, J. S. (2005). High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139.

[163] Xiu, D. and Karniadakis, G. E. (2002a). Modeling uncertainty in steady state diffusion problems via generalized polynomial chaos. *Computer methods in applied mechanics and engineering*, 191(43):4927–4948.

[164] Xiu, D. and Karniadakis, G. E. (2002b). The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24(2):619–644.

[165] Xiu, D. and Karniadakis, G. E. (2003). Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of computational physics*, 187(1):137–167.

[166] Yankov, A. (2015). *Analysis of Reactor Simulations Using Surrogate Models*. PhD thesis, University of Michigan.

[167] Yurko, J. P., Buongiorno, J., and Youngblood, R. (2015). Demonstration of emulator-based bayesian calibration of safety analysis codes: theory and formulation. *Science and Technology of Nuclear Installations*, 2015.

[168] Zhang, E., Feissel, P., and Antoni, J. (2011). A comprehensive bayesian approach for model updating and quantification of modeling errors. *Probabilistic Engineering Mechanics*, 26(4):550–560.