Reduced Order Models for Calibration of Nuclear Reactor Simulation Codes

by

Artem Yankov

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Nuclear Engineering)
in the University of Michigan
2013

Doctoral Committee:

Professor Thomas J. Downar, Chair
Professor John C. Lee
Professor John P. Boyd

2013

This dissertation is dedicated to...

A C K N O W L E D G M E N T S

I would like to thank...

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

PDE partial differential equation

SPDE stochastic partial differential equation

HDMR high-dimensional model representation

ANOVA analysis of variance

TMI Three Mile Island

UAM Uncertainty Analysis in Modeling

PARCS Purdue Advanced Reactor Core Simulator

SCALE Standardized Computer Analyses for Licensing Evaluation

ENDF Evaluated Nuclear Data File

ABSTRACT

Reduced Order Models for Calibration of Nuclear Reactor Simulation Codes

by

Artem Yankov

Chair: Thomas J. Downar

Write abstract here!

CHAPTER 1

Introduction

Introduce the chapter...

1.1 Stochastic Partial Differential Equations

A stochastic partial differential equation (SPDE) is similar in flavor to the better known partial differential equation (PDE), the main difference being the presence of input uncertainties in the former's parameter space. If the parameters in some PDE are probabalistic and the affects of those variable parameters on the outputs are of interest, then the PDE must be reforumulated into a SPDE. The following example aims to elucidate the difference between PDEs and SPDEs and to motivate a discussion of the unique features characteristic to SPDEs. While these features form the mathematical back-bone for this thesis they are somewhat abstract and not crucial to develping an understanding of the subject matter.

Consider the PDE describing one-speed diffusion of neutron flux in a slab nuclear reactor [11]:

$$\begin{aligned} \frac{1}{v}\frac{\partial \phi}{\partial t} - D\frac{\partial^2 \phi}{\partial x^2} + \Sigma_a \phi(x,t) &= \nu\Sigma_f \phi(x,t) \\ \phi(x,0) &= \phi_0(x) \\ \phi(a,t) &= \phi(-a,t) = 0 \end{aligned} \tag{1.1}$$

The PDE in 1.1 can be solved for the flux ϕ as a function of both space x and time t. However, such a solution assumes that the parameters in the PDE, namely v, D, Σ_a and $\nu\Sigma_f$, are fixed values. What would happen to the flux if these parameters were described by probability distributions? The flux would be influenced by perturbations to any of the parameters and so the initial two-dimensional problem is converted to a six-dimensional problem, uncertainty in intial and boundary conditions aside. How would the solution to this SPDE be found? A few preliminaries are in order.

1.1.1 Preliminaries

Most of the language used to describe SPDEs comes from the field of probability theory, which is based on the ideas of sets, fields, and events. The notation and definitions described here come mainly from [31]. The ultimate purpose of introducing the proceeding ideas from probability theory is to be able to understand random variables and the Doob-Dynkin Lemma. Some of the jargon used to describe sets will also be utilized when discussing dimension-wise function decompositions. A *set* is simply a collection of objects while a *subset* is a collection of objects contained within the larger set. A *sample space* is the set of all outcomes of an experiment and is usually denoted by Ω. For example, if the experiment is flipping a fair coin then the sample space is $\Omega = \{H, T\}$. Subsets of Ω are referred to as *events*. If $\Omega = \{\omega_1, \omega_2, ..., \omega_N\}$ then the total number of subsets of Ω is 2^N, where both the empty set $\emptyset$ and all of Ω are included in the count.

A few definitions from set algebra are required before moving on to the Doob-Dynkin Lemma. A *union*, or sum, of two sets A and B is the set of elements that are in at least A or B and is denoted by $A \cup B$. The *intersection* of sets A and B consists of all the elements belonging to both A and B and is denoted by $A \cap B$. Finally, the *complement* of a set A, denoted by A^C, consists of all the elements not in A. From these definition, it follows that $A \cup A^C = \Omega$ and $A \cap A^C = \emptyset$.

Now, let's use the ideas of sets, unions, and intersections to define what is meant by a field and sigma field. Let A and B be subsets of the set Ω. The subsets A and B form a *field* M if,

- $\emptyset \in M$, $\Omega \in M$
- If $A \in M$ and $B \in M$ then $A \cup B \in M$ and $A \cap B \in M$
- If $A \in M$ then $A^C \in M$

A *sigma field* $\mathscr{F}$ is a field that is closed under any countably infinite set of unions and intersections. In other words if subsets $A_1, ..., A_n, ...$ belong to $\mathscr{F}$ then so do $\bigcup_{i=1}^{\infty} A_i$ and $\bigcap_{i=1}^{\infty} A_i$. Consider the sample space $\Omega = \{1, 2, 3\}$. Following the definition of a sigma field, $\mathscr{F} = \{\emptyset, \Omega, \{1\}, \{2, 3\}\}$ is a sigma field while $\mathscr{G} = \{\emptyset, \Omega\{2\}\}$ is not. Rather, the correct sigma field of $\mathscr{G}$ is $\sigma(\mathscr{G}) = \{\emptyset, \Omega, \{2\}, \{1, 3\}\}$. The notation $\sigma(\mathscr{U})$ is used to denote the smallest sigma field containing $\mathscr{U}$, where $\mathscr{U}$ is a collection of subsets of Ω. In general, such a sigma field can be constructed by,

$$\sigma(\mathscr{U}) = \bigcap_{\mathscr{A}} \{\mathscr{U} \subset \mathscr{A} : \mathscr{A} \text{ is a sigma field}\} \tag{1.2}$$

A *Borel sigma algebra* is $\mathscr{B} = \sigma(\mathscr{U})$ where $\mathscr{U}$ consists of all the open sets in $\mathbb{R}^N$.

Combining the ideas described above, a *probability space* $(\Omega, \mathscr{F}, P)$ consists of a sample space Ω, a sigma field $\mathscr{F}$ of subsets of Ω, and a probability function P on $(\Omega, \mathscr{F})$. The probability function must satisfy the three axioms of probability, which are given as:

1. $P(A) \geq 0$
2. $P(\Omega) = 1$
3. $P(A \cup B) = P(A) + P(B)$ if $A \cap B = \emptyset$

An example will help make some of these abstract concepts more concrete. Consider the experiment of tossing a fair coin. The sample space is $\Omega = \{H, T\}$ and the sigma field of events is $\mathscr{F} = \{\{H\}, \{T\}, \Omega, \emptyset\}$. Probabilities of events in $\mathscr{F}$ are $P(H) = P(T) = 1/2$, $P(\Omega) = 1$ and $P(\emptyset) = 0$.

At this point, enough background has been given to describe a *random variable* in general terms. Uncertainty quantification pioneer Gianluca Iaccarino writes, "Random variables are the building blocks for studying uncertainties in a probabilistic framework" [19]. In the simplest terms, a random variable takes events and assigns them a real number. Let X be a random variable. Then X takes an event ω from the event space Ω and maps it to a number on the real line $X(\omega) : \Omega \to \mathbb{R}$. Under such a mapping a whole region $A_B \in \Omega$ gets mapped to an interval B on the real line. A more formal definition of a random variable can be made using the language of set theory. Let $(\Omega, \mathscr{F}, P)$ be a probability space. A mapping $X : \Omega \to \mathbb{R}^N$ measurable with respect to $\mathscr{F}$ is a random variable. In other words, for any open set A in $\mathbb{R}^N$, $X^{-1}(A) \in \mathscr{F}$ is a random variable.

As an example, consider the event space $\Omega = \{1, 2, 3\}$ and the sigma field $\mathscr{F} = \{\emptyset, \Omega, \{1\}, \{2, 3\}\}$. Then if we define $Y(1) = 1$, $Y(2) = 0$, and $Y(3) = -1$ then Y is not a random variable because $\{3\} \notin \mathscr{F}$. Now, if we define $Z(1) = 1$, and $Z(2) = Z(3) = 0$ then Z is a random variable because $\{1\}$ and $\{2, 3\}$ are both in $\mathscr{F}$. This example serves to demonsrate the interconnectedness between sigma fields and event spaces.

The purpose of the preceeding discussion was to build a sufficient framework to be able to state the Doob-Dynkin lemma. Let $X : \Omega \to \mathbb{R}^N$ be a random variable and let $\mathscr{B}$ be the Borel sigma algebra. The sigma algebra generated by X is $\sigma(X) = \left\{X^{-1}(F) : F \in \mathscr{B}\right\}$. The Doob-Dynkin lemma describes the relationship between a random variable and the sigma field it generates. Let $X, Y : \Omega \to \mathbb{R}^N$ be two functions. Then Y is $\sigma(X)$ measurable if and only if there exists a Borel measurable function $g : \mathbb{R}^N \to \mathbb{R}^N$ (for any $A \in \mathscr{B}, g^{-1}(A) \in \mathscr{B}$) such that $Y = g(X)$. To say that Y is "$\sigma(X)$ measurable" means that if X is known then Y is known as well [31]. When a PDE is transformed into a SPDE by treating the parameters

as random variables it is appropriate to question whether the solution of the SPDE can be described in terms of the same random variables [28]. The Doob-Dynkin lemma answers this question with a resounding yes.

1.1.2 Application to Engineering Systems

In this section, the theory of SPDEs is applied to the types of problems that typically arise in nuclear engineering. Much of the notation used in the proceeding discussion is borrowed from [39]. First, define the physical domain as $\mathscr{D} \subset \mathbb{R}^d$ where d can be 1, 2, or 3 depending on the number of spatial dimensions being considered. The boundary of the domain is designated as $\partial\mathscr{D}$. Any coordinate living in the spatial domain can be described by some vector $\mathbf{x} = (x_1, ..., x_d)$. The most general mathematical systems that are of interest can be written as,

$$\mathscr{L}(\mathbf{x}, \omega; u) = f(\mathbf{x}, \omega) \qquad \mathbf{x} \in \mathscr{D} \tag{1.3}$$

$$\mathscr{B}(\mathbf{x}; u) = g(\mathbf{x}) \qquad \mathbf{x} \in \partial\mathscr{D}. \tag{1.4}$$

Ultimately, $u : \Omega \times \mathscr{D} \to \mathbb{R}$ is sought after since u is the solution to 1.3 and 1.4, where $\mathscr{L}$ is a linear/non-linear differential operator, $\mathscr{B}$ is a boundary operator, and f and g are driving terms. The system in 1.3 and 1.4 is defined over a complete probability space $(\Omega, \mathscr{F}, P)$ with $\omega \in \Omega$, as defined previously. Of course, 1.3 and 1.4 must be well-posed in the sense of Hadamard [3]. In Hadamard's definition, well-posed mathematical models of physical phenomena satisfy three conditions:

1. Existence of a solution
2. Uniqueness of the solution
3. The solution is not sensitive to small perturbations in initial conditions

Generally, well-posed problems can be solved using stable computer algorithms. However, it is important not to confuse posedness with conditioning as the two describe very different things. Problems that are typically not well-posed, such as inverse problems, can be formulated as well-posed problems through regularization.

The problem in 1.3 and 1.4 is continuous and as such, will not have an analytic solution for practical problems in engineering applications. An infinite number of random variables are needed to fully describe the stochastic process. Since modeling such a process on a computer is impossible, the infinite-dimensional probability space must be reduced to a finite-dimensional space. The procedure to do this is referred to as the "finite-dimensional

noise assumption". The Karhunen-Loeve expansion of the stochastic space achieves this reduction in dimensionality with the benefit of being able to fully model the full stochastic space if desired [39].

Applying the Karhunen-Loeve expansion to 1.3, the random inputs can be characterized by a set of N random variables as,

$$\mathscr{L}(\mathbf{x}, Y_1(\omega), ..., Y_N(\omega); u) = f(\mathbf{x}, Y_1(\omega), ..., Y_N(\omega)) \tag{1.5}$$

where $\{Y_i(\omega)\}_{i=1}^N$ are uncorrelated random variables. By the Doob-Dynkin lemma, the solution of 1.3 and 1.4 can be expressed in terms of the same random variables $\{Y_i(\omega)\}_{i=1}^N$. Hence, the solution to the SPDE can be written as $u(\mathbf{x}, \omega) = u(\mathbf{x}, Y_1(\omega), ..., Y_N(\omega))$. Equations 1.3 and 1.4 with a discrete stochastic space can be restated as,

$$\mathscr{L}(\mathbf{x}, \mathbf{Y}; u) = f(\mathbf{x}, \mathbf{Y}) \qquad (\mathbf{x}, \mathbf{Y}) \in D \times \Gamma \tag{1.6}$$

$$\mathscr{B}(\mathbf{x}, \mathbf{Y}; u) = g(\mathbf{x}, \mathbf{Y}) \qquad (\mathbf{x}, \mathbf{Y}) \in \partial D \times \Gamma \tag{1.7}$$

where Γ_i is the image of the independent random variables $Y_i(\omega)$. Without loss of generality Γ_i can be restricted to $[0,1]$ for $i = 1, ..., N$. Consequently, the bounded stochastic space, or support, is an N-hypercube $\Gamma = [0,1]^N$. No limits have really been imposed on the stochastic space since any bounded region can be mapped to the unit hypercube. In 1.6 and 1.7 what remains is a set of independent, deterministic equations in space that can be solved using any deterministic discretization technique. The stochastic problem has essentialy been decoupled from the spatial problem.

1.2 Uncertainties in Nuclear Data

1.2.1 Cross Section Uncertainty Overview

While the methods described in this thesis are applicable to a plethora of engineering applications, the target application here concerns the simulation of nuclear reactors. A main component of any reactor analysis is a description of the neutronics, the physical processes behind neutron transport inside a reactor. When studying uncertainty quantification and sensitivity analysis of the neutronics in some reactor system, the primary player has been shown to be uncertainty in cross section data [1] [21]. Cross section uncertainties for various reactions, energies, and nuclides are accumulated during their experimental determination. Many of the cross sections exhibit correlations among themselves that must be taken into account. A problem arises when one becomes interested in performing uncertainty

and sensitivity analysis on a core simulator since the simulators use processed cross section data. The experimental cross section uncertainties and covariances must be carefully propogated down through the standard cross section processing regime.

Until recently, the primary approach for cross section uncertainty propagation involved the application of first-order perturbation theory. The perturbation theory approach is described in detail in [36]. Basically, for neutronics uncertainty analysis this approach entails the solution of an adjoint transport equation for each response of interest, allowing for the efficient retrieval of sensitivity coefficients for all input data. The sensitivity coefficients can then be used to obtain response uncertainties by being operated on by the inputs' covariance matrix. However, the linear approximations in perturbation theory and generalized perturbation theory may fail in certain scenerios. In particular, if the ratio of neutron loss due to absorption is high, such as when a control rod is rapidly inserted, second order effects may become substantial [36]. In addition, while perturbation theory may be computationally efficient for problems where only a handful of responses are of interest, problems with many respones may be burdensome due to the need to solve the generalized adjoint transport equation for each response. Not to mention, implementation of the perturbation theory approach is intrusive to engineering codes which makes uncertainty quantification of coupled, multiphysics code systems infeasible.

With the recent increase in accessibility to parallel computing environments stochastic sampling methods have become popular since they do not apply any kind of approximations to the physics at hand. Consequently, for cross section uncertainty propagation sampling methods have also been adopted to replace perturbation theory. A description of the process for propagating experimental uncertainties in cross section data to few group cross sections is described in this section. The statistical sampling framework for cross section data is extremely flexible since uncertainties can be calculated for any few group parameters. The same cannot be said of perturbation theory, which must formulate responses as ratios of reaction rates. Consequently, uncertanties for few group transport cross sections and assembly discontinuity factors are difficult to obtain using generalized perturbation theory.

1.2.2 Sampling Method

The method for producing stochastically sampled few-group cross sections will be described. Although the method described is quite general, to provide clarity it will be described in the context of the Standardized Computer Analyses for Licensing Evaluation (SCALE) code [6]. Specifically, SCALE's Evaluated Nuclear Data File (ENDF) li-

braries and cross section processing utilities will be detailed. In the ENDF libraries the multigroup cross sections are assumed to be Gaussian and conseqently, they can be fully characterized by their means and covariances. The SCALE 44-group ENDF/B-VII covariance matrix contains generic multigroup covariance data. For a problem where m nuclide-reaction combinations are of interest the pertinent covariance matrix is expanded to a size of $[m \cdot 44] \times [m \cdot 44]$. The Gaussian assumption imposed on the cross section values implies the cross sections can take on negative values. Since cross sections physically cannot take on negative values their distributions are truncated [24].

The first step towards obtaining perturbed few-group cross sections is to sample the generic multigroup covariance library. The covariance data schema in SCALE is given as relative values of infinitely dilute cross sections [38]. Denote the 1D energy dependent, pointwise cross section for reaction x as $\sigma_x(E)$. The infinitely dilute cross section in energy group g can then be calculated as,

$$\sigma_{x,g}(\infty) = \frac{\langle \sigma_x(E) \rangle}{\Delta u_g} \tag{1.8}$$

where the bracket notation indicates a lethargy inner product over the energy interval covered by group g. Consequently, when the multigroup covariance matrix in SCALE is sampled the resulting perturbation $\sigma'_{x,g}$ to the infinitely dilute cross sections can be expressed in terms of a perturbation factor $Q_{x,g}$,

$$\begin{aligned} \sigma'_{x,g}(\infty) &= \left(1 + \frac{\Delta\sigma_{x,g}(\infty)}{\sigma_{x,g}(\infty)}\right) \sigma_{x,g}(\infty) \\ &= Q_{x,g}\sigma_{x,g}(\infty). \end{aligned} \tag{1.9}$$

The perturbations in Eq. 1.9 will provide generic multigroup cross section values to propagate through a transport solver, which can then yield perturbed few-group cross section values. However, to make the perturbed cross sections problem specific they must undergo adjustments due to resonance self shielding. For this purpose, perturbed pointwise cross sections and perturbed Bondarenko self-shielding factors are required. The pointwise, or continuous energy, cross sections are needed to perform 1D transport calculations in the resolved resonance range while the Bondarenko factors are applicable in the unresolved energy range. Since the pointwise cross sections, Bondarenko factors, and generic infinitely dilute multigroup cross section are related algebraically it is necessary to perturb each in terms of the factor $Q_{x,g}$.

In [38] the authors show that, given a perturbation factor $Q_{x,g}$, the appropriate pointwise

cross sections perturbation can be obtained by,

$$\sigma'_x(E) = Q_{x,g}\sigma_x(E) \qquad E \in g. \tag{1.10}$$

From Eq. 1.10 it is clear that the pointwise cross section data gets perturbed uniformly in a given energy group g, which is a valid approximation if the energy width is small. The authors in [38] note that although this approximation leads to discontinuities at the energy group boundaries, it does not impact the resulting multigroup data significantly. No significant impact is expected because the multigroup data is averaged using fluxes and cross sections from the same energy bin.

At this point, infinitely dilute multigroup cross sections and pointwise cross sections can be consistently perturbed using the SCALE ENDF covariance library. To proceed with a transport calculation using perturbed parameters, perturbed Bondarenko self-shielding factors $f'_{x,g}(\sigma_0)$ are also needed. The Bondarenko factors are expressed in terms of the background cross section σ_0, pointwise cross sections, and infinitely dilute cross sections as,

$$f_{x,g}(\sigma_0) = \frac{1}{\sigma_{x,g}(\infty)} \Big\langle \frac{\sigma_x(E)}{\sigma_t(E)+\sigma_0} \Big\rangle \Big/ \Big\langle \frac{1}{\sigma_t(E)+\sigma_0} \Big\rangle. \tag{1.11}$$

Substituting Eqs. 1.9 and 1.10 into Eq. 1.11, the perturbed Bondarenko factor is obtained,

$$f'_{x,g}(\sigma_0) = \frac{1}{Q_{x,g}\sigma_{x,g}(\infty)} \Big\langle \frac{Q_{x,g}\sigma_x(E)}{Q_{t,g}\sigma_t(E)+\sigma_0} \Big\rangle \Big/ \Big\langle \frac{1}{Q_{t,g}\sigma_t(E)+\sigma_0} \Big\rangle. \tag{1.12}$$

The authors in [38] show that Eq. 1.12 can be evaluated by simply evaluating the unperturbed Bondarenko equation in Eq. 1.11 at a perturbed background cross section $\sigma'_0 = \sigma_0/Q_{t,g}$. Note that only those cross sections whose uncertainties are specified in the SCALE covariance library can be processed and propagated through transport calculations. Since covariance data for 2D scattering distributions is not available in the SCALE ENDF/B-VII covariance library, their uncertainty affects cannot be quantified in any analyses [38].

1.2.3 Cross Section Sampling in SCALE

The method described in section 1.2.2 for producing perturbed cross sections will be further described in the context of the modules in SCALE. The first step is to produce perturbation factors using the module XSUSA by sampling the SCALE ENDF/B-VII covariance library. If a total of N_1 output files of transport solver are desired then N_1 sets of perturbation factors are to be obtained. Using the perturbation factors, the SCALE module CLAROL+ yields perturbed, infinitely dilute, multigroup cross sections by essentially applying Eq.

1.9. The resulting cross sections are then passed to the module CRAWDAD+, which outputs perturbed, continuous energy cross sections. To make the multigroup cross sections problem specific they are passed to the BONAMI module, which adjusts the cross sections to account for self-shielding effects in the unresolved energy region using the Bondarenko shielding factor method. Perturbed Bondarenko factors are used in BONAMI, as output from CLAROL+ upon application of Eq. 1.12.

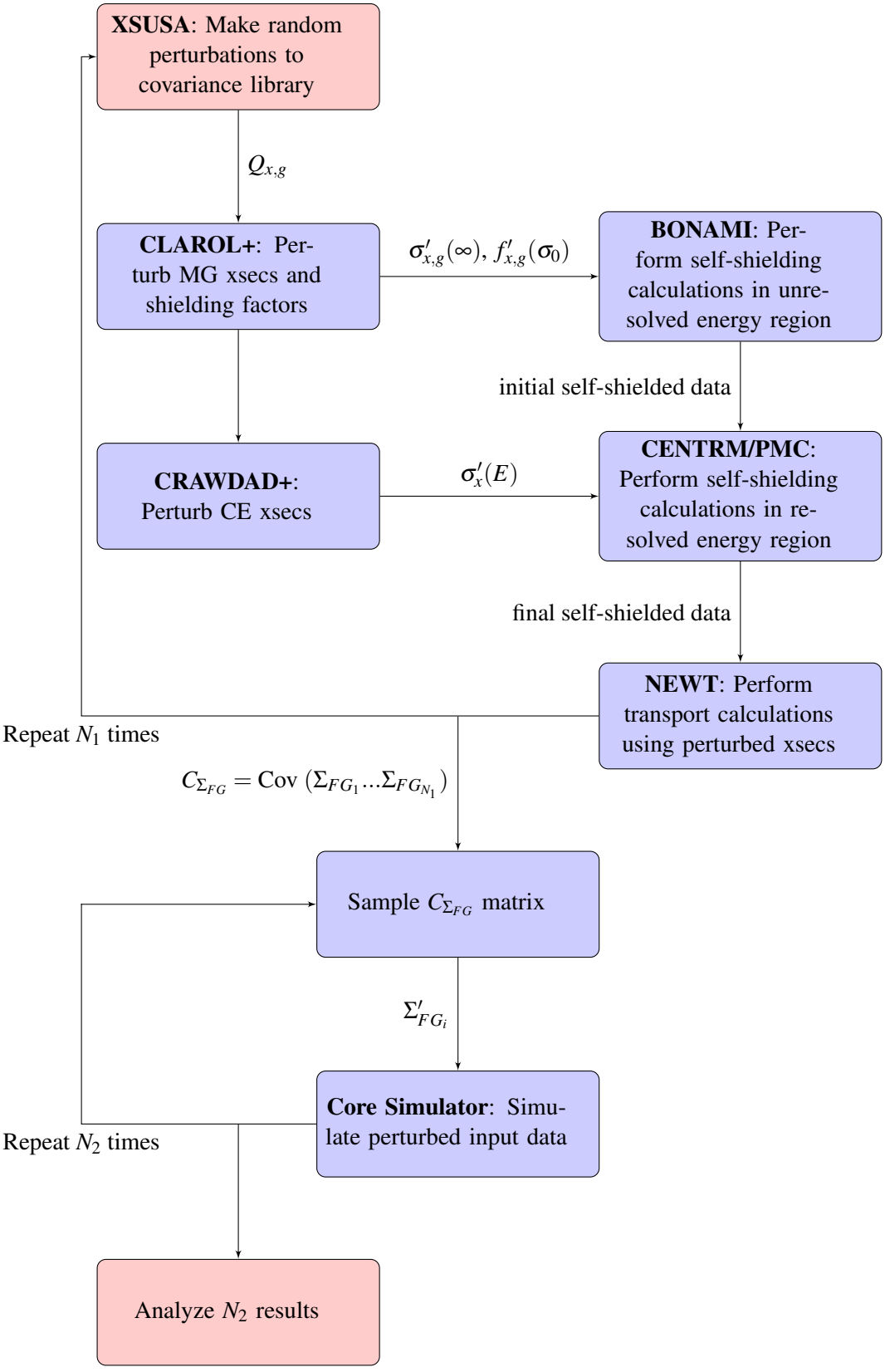
At this point the cross sections still need to be processed for self-shielding effects in the resolved energy region. The SCALE modules CENTRM/PMC are responsible for folding in these effects into the multigroup cross sections. CENTRM solves the neutron slowing down equation on an ultra fine spectra on the order of $\mathcal{O}(10^4)$ energy points. For the flux a linear interpolation scheme is used between energy points to produce a continuous energy solution over the entire energy range [37]. The CENTRM module incorporates actual composition, temperature, and environment dependent effects into the resonance shielded cross sections. PMC accepts the high resolution flux spectra output by CENTRM and averages pointwise nuclear data into multigroup cross sections, which are now tailored to solve the specific problem of interest.

Finally, the problem specific multigroup cross sections output by CENTRM/PMC are sent to a lattice transport solver, which in this case is the SCALE module NEWT. In general, NEWT can be replaced by any transport solver. However, for the purposes of producing few group cross sections a lattice transport solver will generate the multigroup flux distribution for some heterogeneous configuration, which is then used to generate homogenized few group cross sections and intra-assembly flux shapes. A script to extract desireable output quantities from the transport output, such as few group cross sections and kinetics parameters, is needed. Such a script is executed after each of N_1 calls to NEWT. Each call extracts a set of perturbed, few group cross sections that are ultimately propagated through a core simulator. The SAMPLER sequence in SCALE automates the process of producing perturbed cross section sets [38].

With the availability of N_1 perturbed few group cross section sets, each set containing all the nuclear data required by a core simulator, the sets can be directly propagated through the core simulator. The desired results from each simulation can then be gathered and statistically analyzed in what is referred to as the 'one-step' method. However, in what is referred to as the 'two-step' method, the N_1 perturbed few group cross section sets can be used to create a few group covariance matrix [40]. The few group covariance matrix is then sampled and each sample is propagated through the core simulator. Both the 'one-step' and 'two-step' methods have been shown to produce consistent results [40]. However, the 'two-step' method has the advantage of breaking the limitation on the number of samples

that can be propagated through the core simulator. In the 'one-step' method each core simulation requires a corresponding transport solution. Transport solutions are relatively expensive to compute when compared to core simulations. A flow diagram of the 'two-step' method is displayed in Figure 1.1. For the 'one-step' method $N_1 = N_2$ whereas for the 'two-step' method $N_2 >> N_1$, allowing for the acquisition of better statistics.

Figure 1.1: Flow diagram of 'two-step' method for core simulators.



CHAPTER 2

Reduced Order Models for Computer Codes

Computer codes that model physical phenomena typically accept an input file whose purpose is to describe specific conditions in the universe being modeled by the code. The code is executed and the affects of the conditions on some dependent quantities are output. From this perspective computer codes can be viewed and treated in much the same way functions mathematicians deal with are treated. Like matrices, functions with certain properties can undergo various decompositions that offer insight into their structure. The purpose of 2.1 is to describe a technique for decomposing functions into orthogonal components, with the ultimate intention of applying the technique to computer codes. It is hoped that the decomposition of the computer code in terms of its inputs reveals which inputs play the most active roles in the underlying physics. Keeping only the most active dimensions in the decomposition, a reduced order model is effectively built. However, the function decomposition technique described in 2.1 describes only half the story. To create a reduced order model of a computer code that can be efficiently evaluated at any state point in the original parameter space an efficient, multidimensional, interpolation scheme is needed. Such a scheme is discussed in 2.2. The coupling between interpolation and function decomposition that creates a useable reduced order model is described in 2.3.

2.1 Function Decompositions

In order to reduce the dimensionality of some function there must exist a metric to determine the importance of each dimension with respect to the others. It's important to have a framework that isolates the effects of each dimension on the output of the function. The framework chosen to perform dimension reduction is formally known as high-dimensional model representation (HDMR). In statistics, the ANOVA decomposition is a special case of HDMR where the Lesbegue measure is used to perform all integrations.

2.1.1 Dimension-wise Decompositions

The dimension-wise HDMR is algorithmically similar to Gram-Schmidt for matrix orthogonalization in that orthogonal components are systemtically removed to create a linearly independent basis. As in Gram-Schmidt, the essential operator in dimension-wise HDMR is the projection operator. Before introducing the projection operator of interest in this thesis, define the d-dimensional product measure to be,

$$d\mu\left(\mathbf{x}\right) = \prod_{j=1}^{d} d\mu_j\left(x_j\right) \tag{2.1}$$

where μ_j are probability measures defined over some Ω. Two functions $f, g : \Omega^d \to \mathbb{R}$ are considered orthogonal with respect to the product measure defined in 2.1 if the inner product,

$$(f,g) = \int_{\Omega^d} f(\mathbf{x}) g(\mathbf{x}) d\mu(\mathbf{x}) \tag{2.2}$$

is equal to zero. To introduce the projection operator $P_{\mathbf{u}}$, the notation used in [17] is adopted. The operator $P_{\mathbf{u}}$ projects from a d-dimensional space to a $|\mathbf{u}|$-dimensional space for some set $\mathbf{u} \subseteq \mathscr{D}$, where $\mathscr{D} = \{1, ..., d\}$ consists of the set of all coordinate indices in $\mathbf{x}$. The projection on to $\mathbf{u}$ is given as,

$$P_{\mathbf{u}} f\left(\mathbf{x}_{\mathbf{u}}\right) = \int_{\Omega^{d-|\mathbf{u}|}} f\left(\mathbf{x}\right) d\mu_{\mathscr{D}\setminus\mathbf{u}}\left(\mathbf{x}\right) \tag{2.3}$$

where $\mathbf{x}_{\mathbf{u}}$ has length $|\mathbf{u}|$ and consists of the $\mathbf{x}$ coordinates specified in $\mathbf{u}$. Also, the notation $\mathscr{D} \setminus \mathbf{u}$ signifies all the coordinates in $\mathscr{D}$ not contained in $\mathbf{u}$. From 2.3 it is clear that the projection operator works to integrate out all coordinate indices not contained in $\mathbf{u}$ from f. For some coordinate indice sets $\mathbf{u}$ and $\mathbf{v}$, where $\mathbf{u} \neq \mathbf{v}$, the following orthogonality relation holds,

$$(f_{\mathbf{u}}, f_{\mathbf{v}}) = 0. \tag{2.4}$$

The notation $f_{\mathbf{u}}$ is used to denote the function of only the coordinate indices contained in $\mathbf{u}$. From 2.4, it follows that a function can be written in terms of its 2^d orthogonal components,

$$f\left(\mathbf{x}\right) = \sum_{\mathbf{u}\subseteq\mathscr{D}} f_{\mathbf{u}}\left(\mathbf{x}_{\mathbf{u}}\right) \tag{2.5}$$

where the component functions $f_{\mathbf{u}}$ are defined recursively as [17],

$$f_{\mathbf{u}}\left(\mathbf{x}_{\mathbf{u}}\right) = P_{\mathbf{u}} f\left(\mathbf{x}_{\mathbf{u}}\right) - \sum_{\mathbf{v}\subset\mathbf{u}} f_{\mathbf{v}}\left(\mathbf{x}_{\mathbf{v}}\right). \tag{2.6}$$

The recursive definition in 2.6 can be written explicitly as,

$$f_{\mathbf{u}}(\mathbf{x}_{\mathbf{u}}) = \sum_{\mathbf{v}\subseteq\mathbf{u}} (-1)^{|\mathbf{u}|-|\mathbf{v}|} P_{\mathbf{v}} f(\mathbf{x}_{\mathbf{v}}) \tag{2.7}$$

For most functions arising in engineering applications, especially if the function is a computer code, the decomposition in 2.5 is not possible to obtain because each component function $f_{\mathbf{u}}$ will require a high-dimensional integral to be performed. Of course, this statement assumes a Lebesgue measure in the definition of $d\mu$ in 2.1. Alternatively, if a Dirac measure is used then the computationally burdensom integral in 2.3 is reduced to a single function evaluation. In this case, the decomposition in 2.5 is referred to as an anchored-ANOVA decomposition, or CUT-HDMR [29].

2.1.2 Anchored-ANOVA Decomposition

Using the Dirac measure $\delta(\mathbf{x}-\mathbf{a})d\mathbf{x}$ to evaluate the projection operator at a fixed point $\mathbf{a} \in [0,1]^d$ in the hypercube, equation 2.3 becomes,

$$P_{\mathbf{u}} f(\mathbf{x}_{\mathbf{u}}) = f(\mathbf{x})\,|_{\mathbf{x}=\mathbf{a}\setminus\mathbf{x}_{\mathbf{u}}}. \tag{2.8}$$

The notation $\mathbf{a}\setminus\mathbf{x}_{\mathbf{u}}$ is the anchor point $\mathbf{a}$ except at the coordinate indices specified in $\mathbf{u}$. At the coordinate indices $\mathbf{u}$, the anchor point takes upon the corresponding values in $\mathbf{x}$. Using the Dirac measure to evaluate the projections comprising 2.5, the objective function is expressed as a linear combination of its values along lines, faces, hyperplanes,..., etc [17]. As mentioned previously, using the Dirac measure to perform the projection operations in HDMR results in enormous computational savings since high-dimensional integrals are replaced with single function evaluations.

Given the structure of anchored-ANOVA, it is not surprising to learn that there exists a close connection to multivariate Taylor series [26]. This connection provides insight into some of the properties of the anchored-ANOVA decomposition. The multivariate Taylor series of $f(\mathbf{x})$ about a point $\bar{\mathbf{x}}$ can be written as,

$$f(\mathbf{x}) = f(\bar{\mathbf{x}}) + \sum_{i=1}^{d} \frac{\partial f(\mathbf{x})}{\partial x_i}(x_i - \bar{x}_i) + \frac{1}{2!}\sum_{i,j=1}^{d} \frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}(x_i - \bar{x}_i)\left(x_j - \bar{x}_j\right) + \ldots \tag{2.9}$$

As an example, consider what happens if 2.9 is evaluated at $\mathbf{x} = \mathbf{a}\setminus x_i$,

$$f(\mathbf{x})\,|_{\mathbf{x}=\mathbf{a}\setminus x_i} = f(\bar{\mathbf{x}}) + \frac{\partial f(\mathbf{x})}{\partial x_i}(x_i - \bar{x}_i) + \frac{1}{2!}\frac{\partial^2 f(\mathbf{x})}{\partial x_i^2}(x_i - \bar{x}_i)^2 + \ldots \tag{2.10}$$

Since $f_i(x_i) = f(\mathbf{x})|_{\mathbf{x}=\mathbf{a}\setminus x_i} - f(\bar{\mathbf{x}})$,

$$f_i(x_i) = \frac{\partial f(\mathbf{x})}{\partial x_i}(x_i - \bar{x}_i) + \frac{1}{2!}\frac{\partial^2 f(\mathbf{x})}{\partial x_i^2}(x_i - \bar{x}_i)^2 + \ldots \tag{2.11}$$

Expression 2.11 shows that the first-order component functions in anchored-ANOVA consist of entire Taylor series expansions. Similarly, second-order component functions will consist of their respective entire Taylor series expansions and so on. Consequently, a truncated anchored-ANOVA expansion will always provide a better approximation to a function than a truncated Taylor expansion [26].

2.1.2.1 Effective Dimensions

The ultimate purpose of introducing an expansion such as anchored-ANOVA is to truncate it and use the truncated portion as an approximation to the objective function. Of course, evaluation of the truncated anchored-ANOVA expansion is expected to be much more computationally efficient than the objective function. When an anchored-ANOVA decomposition is truncated, the loss incured becomes the components 2.6 that are not being represented. Of course, in practical construction the components not represented are calculated to contribute relatively trivially. Two notions exist for classifying the dimension of a truncated anchored-ANOVA decomposition. Both notions depend on $\hat{\sigma}(f)$, which is the sum of the absolute values of the integrals of all anchored-ANOVA terms [17]

$$\hat{\sigma}(f) = \sum_{\substack{u \subseteq \mathscr{D} \\ \mathbf{u} \neq \emptyset}} |If_{\mathbf{u}}| \approx \sum_{\substack{u \subseteq \mathscr{D} \\ \mathbf{u} \neq \emptyset}} |q_{\mathbf{u}}|. \tag{2.12}$$

The notation $I\cdot$ represents an exact integral but, in practice the integral will be evaluated using some multivariate quadrature scheme and so the exact integral's approximation is denoted by $q_{\mathbf{u}} \approx If_{\mathbf{u}}$. For some user-defined threshold $\alpha \in [0,1]$ the truncation and superposition dimensions of a truncated anchored-ANOVA expansion can be defined. The truncation dimension attempts to quantify the importance of a certain number of dimensions d_t. Mathematically, the truncation dimension is the smallest integer d_t such that,

$$\sum_{\substack{\mathbf{u} \subseteq \{1,\ldots,d_t\} \\ \mathbf{u} \neq \emptyset}} |q_{\mathbf{u}}| \geq \alpha \hat{\sigma}(f).$$

Contrastingly, the superposition dimension attempts to quantify the order of important dimensions d_s. Mathematically, the superposition dimension is the smallest dimension d_s

such that,

$$\sum_{\substack{|\mathbf{u}|\leq d_s \\ \mathbf{u}\neq\emptyset}} |q_{\mathbf{u}}| \geq \alpha\hat{\sigma}(f).$$

Both definitions for the effective definition of a truncated anchored-ANOVA expansion can be directly related to the exact integral of the objective function If. Specifically, for the truncation dimension the following relation holds [17],

$$|If - \sum_{\mathbf{u}\subseteq\{1,\ldots,d_t\}} If_{\mathbf{u}}| \leq (1-\alpha)\,\hat{\sigma}\,(f)\,. \tag{2.13}$$

Similarly, for the superposition dimension the following inequality holds,

$$|If - \sum_{|\mathbf{u}|\leq d_s} If_{\mathbf{u}}| \leq (1-\alpha)\,\hat{\sigma}\,(f)\,. \tag{2.14}$$

Ineqalities 2.13 and 2.14 suggest that if all the anchored-ANOVA terms are used then the exact integral of the objective function can be reproduced. However, in general the set of effective dimensions as determined by anchored-ANOVA will not be equal to the set determined by a classic ANOVA decomposition. The choice of the anchor point **a** has a direct influence on the accuracy and truncation dimension of the anchored-ANOVA expansion [12]. In [12] the authors argue that choosing the anchor point to be the centroid of the parameter space is an excellent choice for most applications. As such, in this thesis the anchor point is always chosen to be the centroid of the working parameter space Ω^d.

2.2 Smolyak Sparse Grids

In order to create a reduced-order model for some objective function the anchored-ANOVA decomposition plays a crucial role but more is needed [13]. Recall that the primary purpose for constructing a reduced-order model is to replace the presumably computationally intensive objective function with something that is trivial to evaluate. Consequently, in evaluating the anchored-ANOVA decomposition at some point **x** the projections in 2.8 must be trivial to evaluate as well. As it stands, evaluating the anchored-ANOVA decomposition for some objective function is significantly more expensive than simply evaluting the function itself. To resolve this issue, a Smolyak sparse grid interpolant is created for each projection. While creating each such interpolant incurs some initial overhead, the payoff is the desired reduced order model.

2.2.1 Motivation

To describe multivariate function interpolation based on Smolyak sparse grids it makes sense to speak in the context of quadrature since a quadrature rule consists of interpolating a function using polynomials and then integrating the polynomials exactly. For the moment consider some smooth 1D function $f(x)$. The function $f(x)$ can be approximated arbitrarily well through the summation,

$$f(x) \approx \sum_{i=1}^{P} f(x_i) C_i(x) \tag{2.15}$$

where $C_i(x)$ are cardinal functions of degree P with the property that $C_i(x_j) = \delta_{ij}$, δ_{ij} being the Kronecker δ-function [7]. By the Weierstrass approximation theorem, smooth functions can be uniformly approximated as closely as desired by polynomial functions [35]. At the collocation points, or abscissas, in 2.15 the function $f(x)$ is interpolated exactly at x_i. The function $f(x)$ is comprised of various constant, linear, quadratic, cubic,..., etc terms and so exact integration of $f(x)$ amounts to integrating its monomial constituents.

Suppose that instead of interpolating a 1D function, a multivariate function of d dimensions is to be interpolated. The naive approach to multivariate interpolation is to take a Cartesian product of 1D rules, such as in 2.15, d times. Consequently, the product grid will contain P^d points, each of which requires a unique function evaluation. Such exponential growth is coined the "curse of dimensionality" [30]. As a rule of thumb, exact integration of a monomial constituent comes at the cost of a single function evaluation [8]. Considering the space of d-dimensional, P-degree polynomials has some,

$$\binom{P+d}{d} \approx \frac{d^P}{P!} \tag{2.16}$$

dimensions, for high dimensional problems the full Cartesian product approach integrates a superfluous number of monomials. Russian mathematician Sergei Smolyak was one of the first to realize the potential computational savings in his paper [33].

2.2.2 Algorithm Mechanics

A Smolyak sparse grid is the set of collocation points used to build an interpolant for some multivariate objective function while the Smolyak algorithm is the whole procedure of building the interpolant. To begin, the Smolyak algorithm will be stated and pertinent notation will be introduced. Since indice tracking comprises the brunt of understanding the Smolyak algorithm, it is crucial to choose a clear notation. Consequently, the notation used here closely follows that of [4].

Slightly generalizing 2.15, for the case of some smooth 1D function f, let U^i be the interpolant of f comprised of m_i collocation points.

$$U^i = \sum_{j=1}^{m_i} f\left(x_j^i\right) a_j^i \tag{2.17}$$

In 2.17, $i \in \mathbb{N}$, and $a_j^i \in C([-1,1])$ are basis functions imposing the demand that U^i exactly be able to reproduce f at the collocation points x_j^i. The notation $x_j^i \in [-1,1]$ refers to the j^{th} collocation point of m_i total points. Restricting the domain of the collocation points to $[-1,1]$ does not impose any limitations on being able to interpolate f arbitrarily well since $[-1,1]$ can always be mapped to the parameter space of f.

To generalize from 1D interpolation to multivariate interpolation 1D interpolation formulas, such as the one in 2.17, are combined using tensor products.

$$\left(U^{i_1} \otimes \cdots \otimes U^{i_d}\right)(f) = \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} f\left(x_{j_1}^{i_1}, \cdots, x_{j_d}^{i_d}\right) \cdot \left(a_{j_1}^{i_1} \otimes \cdots \otimes a_{j_d}^{i_d}\right) \tag{2.18}$$

Tensor products are a mathematical convenience used to represent all combinations of some entity, in this case U^i. The scheme in 2.18 suffers from the, "curse of dimensionality" since a total of,

$$\prod_{k=1}^{d} m_{i_k} \tag{2.19}$$

function evaluations are needed to form the interpolant. The Smolyak algorithm is based on 2.18, the only difference being not all the tensor products are used. In explicit form, the Smolyak formula for approximating the left-hand side of 2.18 is given as [4],

$$A(q,d) = \sum_{q-d+1 \leq |\mathbf{i}| \leq q} (-1)^{q-|\mathbf{i}|} \binom{d-1}{q-|\mathbf{i}|} \left(U^{i_1} \otimes \cdots \otimes U^{i_d}\right). \tag{2.20}$$

Each entry i_k in the vector $\mathbf{i} \in \mathbb{N}^d$ contains the indice corresponding to the level of interpolation in dimension k. The more collocation points being utilized, the higher the level of interpolation since the interpolant becomes increasingly accurate. The magnitude of $\mathbf{i}$ is $|\mathbf{i}| = |i_1 + \cdots + i_d|$. Since each $i_d \geq 1$, the variable $q \geq d$. The variable q essentialy keeps track of the level of interpolation of the Smolyak algorithm. As q is increased, more tensor product combinations are allowed. From 2.20 it is clear that the Smolyak algorithm is able to reduce the total number of tensor product components by limiting the entries of $\mathbf{i}$.

The Smolyak formula in 2.20 can be rewritten in several ways, all of which use the idea

of the incremental interpolant Δ^i defined as,

$$\begin{aligned} U^0 &= 0 \\ \Delta^i &= U^i - U^{i-1} \end{aligned} \tag{2.21}$$

The incremental interpolant operator is simply the difference between interpolants at two succesive levels. Using the notion of the incremental interpolant, the Smolyak formula can be rewritten as,

$$A(q,d) = \sum_{|\mathbf{i}|\leq q} \left(\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d}\right) \tag{2.22}$$

At first sight, 2.20 and 2.22 seem inefficient since neither exposes the recursiveness inherent in the Smolyak formula. In other words, when moving from index q to $q+1$ the work done to get to level q is not lost. Rewriting the Smolyak formula in a recursive fashion is advantageous for implemention on a computer.

$$A(q,d) = A(q-1,d) + \Delta A(q,d) \tag{2.23}$$

$$\Delta A(q,d) = \sum_{|\mathbf{i}|=q} \left(\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d}\right) \tag{2.24}$$

While the Smolyak algorithm representation in 2.23 has the advantage of being represented recursively, it does not provide any type of indicator for when the Smolyak sparse grid should be refined. Collocation points should be added to a Smolyak sparse grid until the resulting interpolant is able to reproduce the objective function to some user-defined threshold. The authors in [28] are able to rewrite 2.24 in terms of what's referred to as a hierarchical surplus,

$$\Delta A(q,d) = \sum_{|\mathbf{i}|=q} \left(f(x_{j_1}^{i_1},...,x_{j_d}^{i_d}) - A(q-1,d)(x_{j_1}^{i_1},...,x_{j_d}^{i_d})\right) \cdot \left(a_{j_1}^{i_1} \otimes \cdots \otimes a_{j_d}^{i_d}\right) \tag{2.25}$$

which appears as the first term in the summation as the difference between the function value at the point $(x_{j_1}^{i_1},...,x_{j_d}^{i_d})$ and the Smolyak $q-1$ level interpolant value at the same point. Level q of the Smolyak algorithm generally contains all the points comprising level $q-1$ plus some new collocation points. Consequently, the level q Smolyak interpolant is expected to exactly evaluate any collocation points born in previous levels. The summation in 2.25 is taken over all the new points in level q that have not appeared in level $q-1$ since the hierarchical surplus for these will be identically equal to zero. The hierarhchical surpluses provide an indicator for how well the Smolyak algorithm is interpolating some objective function. If the hierarchical surpluses are decreasing with each successive level

then the Smolyak algorithm is converging.

Following the notation in [4], let $X^i = \{x_1^i, ..., x_{m_i}^i\}$ be the collocation points comprising U^i. From 2.20, the total number of collocation points in a Smolyak sparse grid can be written as,

$$H(q,d) = \bigcup_{q-d+1 \leq |\mathbf{i}| \leq q} \left(X^{i_1} \times \cdots \times X^{i_d}\right). \tag{2.26}$$

2.2.3 Basis and Collocation Points

The exactness of the Smolyak algorithm is decided mainly by the choice of collocation points $x_{j_k}^{i_k}$ used to build $H(q,d)$. The basis functions $a_{j_k}^{i_k}$ work to weave the collocation points together. Gaussian quadrature is a favorite of many since with only $n+1$ collocation points, all polynomials of degree $2n+1$ or less can be integrated exactly [16]. However, collocation points derived from Gaussian quadrature schemes are not nested in that $X^i \not\subset X^{i+1}$. Nestedness in the choice of collocation points is an essential feature for reducing the computational expense of applying the Smolyak algorithm. If nested collocation points are chosen for each X^{i_k} then the Smolyak sparse grid will also be nested such that $H(q-1,d) \subset H(q,d)$ [4]. Consequently, when improving the Smolyak interpolant from level $q-1$ to level q one will only have to evaluate the objective funtion at the points that are unique to X^i, which are given as $X_\Delta^i = X^i \setminus X^{i-1}$ [28]. The set of new points in level q of a Smolyak sparse grid are given as,

$$\Delta H(q,d) = \bigcup_{|\mathbf{i}|=q} X_\Delta^{i_1} \times \cdots \times X_\Delta^{i_d}. \tag{2.27}$$

A viable alternative to Gaussian quadrature collocation points for the Smolyak algorithm is to use Clenshaw-Curtis collocation points, which consist of the extrema of Chebyshev polynomials. While $n+1$ Clenshaw-Curtis abscissas can only exactly integrate polynomials of degree n, they have the advantage of being nested. Accuracy is sacrificied for nestedness in the Smolyak algorithm, at least in theory. In practice it has been shown that for most functions Clenshaw-Curtis quadrature performs almost on par to Gaussian quadrature [34]. In other words, the double accuracy of Gaussian quadrature is rarely realized. For some level i the Clenshaw-Curtis collocation points are given by,

$$x_j^i = \begin{cases} \cos\frac{\pi(j-1)}{m_i-1} & j = 1, ..., m_i \text{ if } i > 1 \\ 0 & j = 1 \text{ if } i = 1 \end{cases} \tag{2.28}$$

In order for the level i Clenshaw-Curtis abscissas to contain the level $i-1$ abscissas, a total of 2^{i-1} new points must be added. Consequently, the total number of abscissas appearing

in the level i Clenshaw-Curtis scheme is given as,

$$m_i = \begin{cases} 2^{i-1}+1 & i > 1 \\ 1 & i = 1 \end{cases} \tag{2.29}$$

Another alternative to the Gaussian and Clenshaw-Curtis abscissas is Gauss-Patterson. The Gauss-Patterson set of collocation points are nested and provide a polynomial exactness of $(3n-1)/2$ with n points, which is right in between the exactness of Clenshaw-Curtis and Gaussian sets. Obtaining the Gauss-Patterson abscissas involves a rather convoluted, iterative process and so the reader is referred to [9] to review the methodology and obtain tables of the actual points. The growth rule for Gauss-Patterson goes as $2^i - 1$, which is some factor of two greater than the growth rule for Clenshaw-Curtis. In [27], the authors conclude the Gauss-Patterson collocation points are competitive with Clenshaw-Curtis when comparing the cost and accuracy of computing quadratures using the same number of function evaluations.

To weave together the collocation points forming a Smolyak sparse grid, some type of basis function a_j^i is needed, as defined in 2.17. Although the basis functions will be applied to multi-dimensional interpolation, the Smolyak algorithm conveniently scales 1D basis functions to multiple dimensions through the use of tensor products. One basis commonly used in adaptive sparse grids is the linear hat basis function [2]. For the scheme in 2.29 the linear hat functions are given as,

$$\begin{aligned} a_1^1 &= 1 \text{ for } i = 1, \\ a_j^i &= \begin{cases} 1-(m_i-1)|x-x_j^i| & \text{if } |x-x_j^i| < 1/(m_i-1) \\ 0 & \text{else} \end{cases} \end{aligned} \tag{2.30}$$

for $i > 1$ and $j = 1, ..., m_i$. While the linear hat functions have the advantage of local support they are limited to relatively slow convergence due to their lack of curvature. Offering faster error decay are the global Lagrange characteristic polynomials,

$$a_j^i = \begin{cases} 1 & \text{if } i = 1 \\ \prod\limits_{\substack{k=1 \\ k \neq j}}^{m_i} \dfrac{x - x_k^i}{x_j^i - x_k^i} & j = 1, ..., m_i \text{ for } i > 1 \end{cases} \tag{2.31}$$

However, the Lagrange charcteristic polynomials are plagued by the fact that each evaluation of 2.17 requires $\mathcal{O}(m_i^2)$ operations and often the computation is numerically unstable [5]. To remedy these concerns, the barycentric form of Lagrange characteristic polyno-

mials is used to form a basis. The barycentric Lagrang basis is given as,

$$a_j^i = \begin{cases} 1 & \text{if } i = 1 \\ \dfrac{\frac{w_j^i}{x - x_j^i}}{\sum_{j=0}^{m_i} \frac{w_j^i}{x - x_j^i}} & j = 1, ..., m_i \text{ for } i > 1 \end{cases} \tag{2.32}$$

where w_j^i are barycentric weights defined by,

$$w_j^i = \frac{1}{\prod_{k \neq j} (x_j^i - x_k^i)} \qquad j = 1, ..., m_i. \tag{2.33}$$

For special collocation sets, such as Clenshaw-Curtis in 2.28, explicit forms exist for the barycentric weights. Generally, forming the weights is an $\mathcal{O}(m_i^2)$ operation and then evaluation of an interpolant based on the barycentic Lagrange basis is only a $\mathcal{O}(m_i)$ operation [5]. With an explicit form in hand, evaluation of the barycentric Lagrange basis is significantly cheaper than the Lagrange basis. For the Clenshaw-Curtis collocation points in 2.28, the barycentric weights are given by [32],

$$w_j^i = (-1)^{j+1} \delta_j^i \qquad \delta_j^i = \begin{cases} .5 & j = 1 \text{ or } j = m_i \\ 1 & \text{else} \end{cases}. \tag{2.34}$$

From 2.32, an apparent problem exists if the barycentric basis is to be evaluated at a collocation point. As [5] explains, the problem can be circumvented by simply perturbing the value of x by an ε on the order of machine precision. In this case, the numerator and denominator in 2.32 will effectively cancel each other such that $a_j^i = 1$. The barycentric Lagrange basis is therefore numerically stable.

2.2.4 Exactness and Error Bounds

The exactness of the Smolyak algorithm is determined by the space of polynomials the algorithm is exact on. Since the 1D interpolation rules, on which the Smolyak algorithm is based on, can exactly interpolate certain polynomials it is not presumptuous to expect the Smolyak algorithm to exactly interpolate certain polynomial spaces. Using the collocation set in 2.29 and 2.28 the Smolyak interpolant $A(q,d)$ is exact on [4],

$$\sum_{|\mathbf{i}|=q} \mathbb{P}(m_{i_1} - 1, 1) \otimes \cdots \otimes \mathbb{P}(m_{i_d} - 1, 1) \tag{2.35}$$

where $\mathbb{P}(k,d)$ is the space of all polynomials in d dimensions of total degree no greater than k. From 2.35 it follows that the Smolyak interpolant for $q = d + P$ is exact for all polynomials of degree P. In other words, the effects of any monomials containing x^l for $l \leq P$ will be captured by the Smolyak algorithm. Recall from 2.16 that the degrees of freedom of $\mathbb{P}(P,d)$ goes as $d^P/P!$. Any method aiming to reproduce $\mathbb{P}$ requires at least this many function evaluations. Since the number of collocation points in a Smolyak grid for $A(d+P,d)$ goes as $2^P d^P/P!$ the dependence on dimension is said to be optimal [4]. However, the asymptotic growth of points also indicates that the Smolyak algorithm requires still excessive function evaluations to achieve polynomial exactness.

Since the Smolyak algorithm is constructed using one-dimensional interpolation formulas, which all have error bounds, it is also possible to derive error bounds for a Smolyak interpolant $A(q = d + P,d)$. While the reader is instructed to consult [4] for a detailed derivation of the error bounds, they will nevertheless be stated here. Consider some d-variable function f with continuous derivatives of order P in each variable. The error in using a Smolyak interpolant to approximate f can be given as,

$$\|f - A(d+P,d)(f)\|_\infty \leq c_{d,P} M^{-P} (\log M)^{(P+2)(d-1)+1} \tag{2.36}$$

where M is the total number of knots used by $A(d+P,d)$ and $c_{d,P}$ is a constant depending on both d and P. From 2.36, the error in the Smolyak interpolant heavily depends on the smoothness of the function being interpolated and on the total number of collocation points used to form the interpolant.

2.2.5 Computer Implementation

To implement Smolyak's algorithm on a computer equations 2.23 and 2.25 should be utilized since together they provide a recursive definition. Much of the implementation efforts are concerned with indice book keeping. Although the pseudocode for Smolyak's algorithm used in this thesis is provided here, the reader is directed to [25] for more elaborate details. Efficiency of the algorithm can be increased by pre-calculating the desired abscissas as in 2.28, the number of ascissas at a given level as in 2.29, and corresponding barycentric weights as in 2.34. A data structure for quick retrieval of the desired values is also necessary. Abscissa information is constantly being reused in Smolyak's algorithm and so it is inefficient to have to recalculate values each time.

The pseudocode for Smolyak's algorithm in algorithm 1 will now be discussed in some detail. To initialize the algorithm a data structure must be created to store all information for each index in the sparse grid. For level q of the Smolyak algorithm the summation in 2.23

Algorithm 1 Smolyak's algorithm for creating an interpolant for a function f of d dimensions. The algorithm will exit if the maximum Smolyak level is reached or if one of the convergence criteria is met.

```
Create data structure that stores indice coordinates and hierarchical surplus.
for q = 0, maximum level do
    for (i_1, ..., i_d) in enumerations of i_1 + ... + i_d = q + d do        ▷ See Alg. 2
        for (j_1, ..., j_d) in enumerations of (i_1, ..., i_d) do           ▷ See Alg. 3
            Turn each (j_1, ..., j_d) into a knot.
            Categorize as either processed or unprocessed knot.
        end for
        Evaluate f(unprocessed knots).
        Calculate hierarchical surplus at unprocessed knots.                ▷ Eq. 2.25
        Archive newly processed knots.
    end for
    Check for convergence.
end for
```

is over all sets $(i_1, i_2, ..., i_d)$ such that $i_1 + i_2 + ... + i_d = q + d$. Each such set corresponds to a knot $(x_{j_{i_1}}^{i_1}, ..., x_{j_{i_d}}^{i_d})$ in the random space defined by the hypercube $[-1, 1]^d$. The knot, function value at the knot, and the corresponding hierarchical surplus should all be stored in the data structure.

The first loop in the pseudocode tells the code to keep increasing the interpolation level in the Smolyak algorithm until some maximum level is reached, which is specified by the user. The purpose of this loop is to make sure the algorithm ends eventually. Of course, other convergence criteria are in place in hope that the algorithm terminates long before the maximum level is reached. The second loop goes through all the enumerations of $(i_1, i_2, ..., i_d)$ such that $i_1 + i_2 + ... + i_d = q + d$. The algorithm for producing such enumerations is provided in algorithm 2. The third loop takes each enumeration and again enumerates over each index to obtain each component in the tensor product appearing in the Smolyak formulation. An algorithm to execute this enumeration is provided in algorithm 3.

In the main body of the pseudocode each output from algorithm 3 is first converted to a knot $(x_{j_{i_1}}^{i_1}, ..., x_{j_{i_d}}^{i_d})$. Each potential knot must then be sorted into one of two categories. The motivation for the two categories arises from the fact that the same knot may be expressed in several ways. Since each knot corresponds to a function value and hierarchical surplus, significant computational savings can be incurred by not reevaluating the objective function at the same knots. Consequently, each potential knot is binned into either a category of knots that have already been evaluated at f or a category of unevaluated knots.

Once all components in the tensor product for a given $(i_1, i_2, ..., i_d)$ have been converted

and sorted, the unevaluated knots are processed. In this step of the algorithm the previously unevaluated knots should be evaluated at f in parallel if possible since each evaluation is completely independent. The resulting functions values should then be used to compute the hierarchical surplus in 2.25 for each knot. Once the function value and hierarchical surplus is available for each new knot the results should be archived in the indice data structure.

Finally, once the second loop is complete, the level of the Smolyak interpolant has been effectively increased and it's time to check whether additional levels are required based on user-defined convergence criteria. Perhaps the best indicator of a Smolyak interpolant's convergence is the maximum hierarchical surplus calculated for all newly processed knots at the current interpolation level. The hierarchical surplus is a measure of how well the interpolant is able to match the objective function and therefore, if the hierarchial surpluses being calculated are decreasing with each level the interpolant is converging. An additional convergence criteria includes comparison of the relative change in computed mean and variance between two successive interpolation levels. For this thesis, the Smolyak algorithm is terminated only after the maximum hierarchical surplus is below a certain threshold, the relative change in interpolant mean is below a threshold, and the relative change in variance does not exceed a threshold.

2.3 Combining Decomposition and Smolyak's Algorithm

As hinted at in the beginning of chapter 2, the Smolyak algorithm combines with the anchored-ANOVA decomposition to create a reduced order model for any well behaving computer code. To see how the Smolyak algorithm fits into the functional decomposition described in this chapter, begin by substituting 2.7 into 2.5 to get,

$$f(\mathbf{x}) = \sum_{\mathbf{u}\subseteq\mathscr{D}} \sum_{\mathbf{v}\subseteq\mathbf{u}} (-1)^{|\mathbf{u}|-|\mathbf{v}|} P_{\mathbf{v}} f(\mathbf{x}_{\mathbf{v}}). \tag{2.37}$$

Now, insert the Dirac projection operator from 2.8 into 2.37 to arrive at,

$$f(\mathbf{x}) = \sum_{\mathbf{u}\subseteq\mathscr{D}} \sum_{\mathbf{v}\subseteq\mathbf{u}} (-1)^{|\mathbf{u}|-|\mathbf{v}|} f(\mathbf{x})|_{\mathbf{x}=\mathbf{a}\setminus\mathbf{x}_{\mathbf{v}}}. \tag{2.38}$$

To create a reduced order model the set $\mathscr{D}$ is ultimately shrunk to only contain a subset of all the variables of f but this is discussed later. The important aspect of 2.38 to realize is that in order to evaluate $f(\mathbf{x})|_{\mathbf{x}=\mathbf{a}\setminus\mathbf{x}_{\mathbf{v}}}$ evaluation of the expensive computer code $f(\mathbf{x})$ is required. Consequently, the desireable property of reduced order models, that of rapid evaluation, is not achieved in 2.38. The remedy is to approximate each $f(\mathbf{x})|_{\mathbf{x}=\mathbf{a}\setminus\mathbf{x}_{\mathbf{v}}}$ using

Smolyak interpolants. Substituting 2.22 into 2.38, the formulation for creating a reduced order model of f is complete.

$$f(\mathbf{x}) = \sum_{\mathbf{u}\subseteq\mathscr{D}} \sum_{\mathbf{v}\subseteq\mathbf{u}} (-1)^{|\mathbf{u}|-|\mathbf{v}|} \sum_{|\mathbf{i}|\leq q} \left(\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_{|\mathbf{v}|}}\right) \tag{2.39}$$

While there is initial overhead to create an interpolant for each component in 2.38 the result is quick evaluation of the reduced order model. Details regarding the implementation and application of 2.39 will be discussed in the proceeding sections.

2.3.1 Combinatorics Routines

In order to implement 2.39 on a computer several enumeration routines need to be available. Unfortunately, these routines are not available in most numerical math libraries containing combinatorics routines. The first routine of interest solves the problem of how to enumerate all the ways d positive integers can be summed to equal another integer. In other words, what are all the sets $\{i_1,...,i_d\}$ such that $i_1+i_2+...+i_d = q$? This problem inserts itself in 2.39 in the summation index for Smolyak interpolation. As the Smolyak interpolant becomes refined from level to level—q is increased by one in each refinement—the Smolyak algorithm must newly account for all $\mathbf{i}$ such that $|\mathbf{i}| = q$. The following enumeration algorithm, a slight modification of the original algorithm found in [17], finds all the desired indice sets:

With all the index sets for some Smolyak level q available through the code segment in 2, the tensor product appearing in 2.22 can be evaluated with the aid of an additional enumeration routine. Each indice in an index set $\{i_1+i_2+...+i_d\}$ corresponds to certain number of knots, which for example, is given by 2.29 for Clenshaw-Curtis. All the components in the tensor product can be given by the following enumeration algorithm, which is based on the algorithm in [17]. Input to algorithm 3 should be based on output from algorithm 2. Specifically for some index set $\{i_1,i_2,...,i_d\}$ returned by algorithm 2, each indice should be converted to a corresponding number of knots and input to algorithm 3.

2.3.2 Sampling Sparse Grid Interpolant of Correlated Variables

The reduced order model in 2.39 consists of linear coombinations of Smolyak interpolants. Evaluation of the reduced order model is equivalent to finding the value at several Smolyak interpolants and summing the results. However, recall that in the Smolyak algorithm for building the interpolants described in 2.2.2 each dimension in the sparse grid is built orthogonal to the others. Implicit in this construction is the assumption that the random variables

Algorithm 2 For postitive integers d and q this code outputs all sets $\{i_1, i_2, ..., i_d\}$ such that $i_1 + i_2 + ... + i_d = q$.

$p = 0$
$m = q - d + 1$
$k = [0, 1, ..., 1]$ ▷ vector of length d
$\hat{k} = [m, m, ..., m]$ ▷ vector of length d
repeat
    $k(p) = k(p) + 1$
    if $k(p) > \hat{k}(p)$ **then**
        if p=d **then**
            All indices enumerated!
        else
            $k(p) = 1$
            $p = p + 1$
        end if
    else
        for $j = 0 : p$ **do**
            $\hat{k}(j) = \hat{k}(p) - k(p) + 1$
        end for
        $k(0) = \hat{k}(0)$
        $p = 1$
        Return valid index set k!
    end if
until $k = [0, ..., 0]$

Algorithm 3 Code for enumerating all components of a tensor product. The input is a d dimensional vector m where each entry m_j corresponds to the number of knots in a collocation scheme of level i_j.

$p = 0$
$s = [0, 1, 1, ..., 1]$ ▷ vector of length d
repeat
    $s(p) = s(p) + 1$
    if $s(p) > m(p)$ **then**
        if $p = d - 1$ **then**
            All indices enumerated!
        else
            $s(p) = 1$
            $p = p + 1$
        end if
    else
        $p = 0$
        Return valid enumeration set!
    end if
until $s = m$

comprising the objective function are independent. In fact, in many computer codes the random variables forming the parameter space are correlated. The degree of correlation among the random variables is generally described using a covariance matrix.

With the availability of a covariance matrix for the input random variables it is possible to sample the reduced order model hundreds or thousands of times to get accurate and precise statistical moments. Evaluating a Smolyak interpolant is relatively cheap and fast so this is not a computational problem. To produce correlated random variables from a Gaussian population the Kaiser-Dichman method can be applied [23]. Say some inputs to a computer code are normally distributed with mean μ and covariance Σ. To produce a random sample μ' from Σ one can apply,

$$\mu' = \mu + U^T \pi \tag{2.40}$$

where U^T is the lower triangular matrix arising from the Cholesky decomposition of Σ and π is a standard normal random vector.

The Cholesky decomposition of a covariance matrix $C = U^T U$ is the equivalent to taking the square root of a matrix. A Cholesky matrix transform, or left multiplication by U^T, maps uncorrelated variables into correlated variables with covariance matrix Σ. Consequently, in 2.40 μ' is normally distributed with mean μ and covariance Σ. A statistically significant number of instances of μ' should be calculated and evaluated at the reduced order model. Even though the model is built assuming independent variables this method takes into account any correlations when calculating statistical moments. When evaluating any Smolyak interpolant at a number of sampled points one must always make sure all the sampled points lay inside the bounds of the hypercube forming the sparse grid.

2.3.3 Dimension Truncation

By definition a reduced order model contains less dimensions than the original model whose reduction is intended. Consequently, a methodology for identifying important dimensions is necessary. For the purposes of this thesis, the importance of a random input variable on some objective function is determined by its contribution to the function's variance. While there are methods for exactly calculating the truncation and superposition dimensions of a function [17], these methods require the computation of all 2^d component functions in an ANOVA decomposition. For typical computer codes used in engineering this requirement is not feasible and so the effective dimensions of a function must be estimated adaptively.

To this end, construction of a reduced order model based on the anchored-ANOVA

decomposition begins with calculation of the zeroth-order and all first-order components. The zeroth-order component function is simply the function evaluated at the anchor point,

$$f_{\{\emptyset\}} = f(\overline{\mathbf{x}}). \tag{2.41}$$

From the recursive definition of the anchored-ANOVA decomposition in 2.6, the i^{th} component function is given as,

$$f_{\{i\}} = f(\mathbf{x})|_{\mathbf{x}\setminus x_i} - f_{\{\emptyset\}}. \tag{2.42}$$

The first order components solely measure the affect of the i^{th} random variable on the function output. Therefore, for each of d random variables contributing to a function's variability a sensitivity coefficient can be calculated as [29],

$$\eta_i = \frac{\int_{\Omega_i} \left[f(\mathbf{x})|_{\mathbf{x}=\overline{\mathbf{x}}\setminus x_i} - f(\mathbf{x}) \right] \rho(x_i) dx_i}{f_{\{\emptyset\}}} \tag{2.43}$$

where $\rho(x_i)$ is the probability density of x_i. If f is a function of spatial coordinates then the L_2 norm can be applied to 2.43. The greater the affect of the i^{th} random variable on the function output, the greater the sensitivity coefficient.

Once all the first order anchored-ANOVA components are calculated the sensitivity coefficients in 2.43 can be used to identify the important dimensions. To start, each η_i should be normalized by the sum of all η_i. The important dimensions can then be obtained by taking all i variables such that $\eta_i > \theta_1$. A value of $\theta_1 = .02$ has been shown to be effective for many problems [13]. At the very least, the reduced order model described here will consists of the zeroth order and all first order anchored-ANOVA components, giving it a superposition dimension of one. The variance and mean of the reduced order model with superposition dimension one should be obtained through sampling using 2.40.

To improve accuracy higher order components can be included. Higher order components should only be built using combinations of the dimensions deemed important. If all dimensions were to be included construction of a reduced order model based on the ANOVA decomposition would quickly become intractable for most engineering problems. It can be argued that if a set of random variables independently affect a function then their interactions will likely also affect the function. Say all first order components have been constructed and the important dimensions have been added to a set $\mathscr{T}$. The next step in creating a higher order model is to construct anchored-ANOVA components utilizing dimensions t such that $t \subset \mathscr{T}$ and $|t| = 2$. In total there should be $C(|\mathscr{T}|, 2)$ such components.

In general, the number of p order components can be given as,

$$C(|\mathscr{T}|, p) = \frac{|\mathscr{T}|!}{(p!)(|\mathscr{T}| - p)!}. \tag{2.44}$$

To determine whether additional orders need to be included in the reduced order model, the mean of the current reduced order model should be compared to the mean of the previous order model. If the relative change in mean doesn't exceed some threshold θ_2 then the reduced order model is considered converged. Otherwise, higher order components should be added. In this manner, a reduced order model for a computer code can be constructed adaptively. The procedure for adaptively creating a reduced order model is summarized in algorithm 4.

Algorithm 4 Adaptively creates a reduced order model for some function f of d dimensions.

```
Create zeroth-order component function.                          ▷ Eq. 2.41
for i = 1, d do
    Construct 1D Smolyak interpolant for P_{i} f.                ▷ Eq. 2.8
    Create first-order anchored-ANOVA component f_{i}.           ▷ Eq. 2.42
end for
Identify important dimensions and put into set T.                ▷ Eq. 2.43
for p = 2, d do
    for t ⊂ T s.t. |t| = p do
        Construct p-dimensional Smolyak interpolant for P_t f.   ▷ Eq. 2.8
        Create p-order anchored-ANOVA component f_t.             ▷ Eq. 2.6
    end for
    Check for convergence of reduced order model.
end for
```

CHAPTER 3

Application to Reactor Systems

The methodology described in Chapter 2 will initially be applied to three relatively simple problems in reactor physics, each building on its predecessors in complexity. The purpose of the simple problems is to develop some untuition about the workings of the described reduced order model algorithm. Ultimately, the methods will be applied to solve a state of the art problem in reactor uncertainty quantification and sensitivity analysis.

3.1 Infinite Lattice Multiplication Factor

3.1.1 Problem Statement

In an infinite lattice a reactor of infinite size is considered and therefore neutrons are not capable of leaking out of the system. An infinite lattice effectively removes the effects of geometry in neutron transport and characterizes the system entirely in terms of its material properties. Since the physics of an infinite lattice are greatly simplified an analytic analysis of the system is possible. Consequently, the infinite lattice problem is ideal for an initial analysis of any new computational method.

To begin, the mathemtical formulation of an infinite lattice will be described. In matrix form the two-group neutron balance equations for an infinite lattice can be written as,

$$\begin{pmatrix} \Sigma_{a_1}+\Sigma_{1\to 2} & 0 \\ -\Sigma_{12} & \Sigma_{a_2} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} = \frac{1}{k_\infty} \begin{pmatrix} \nu\Sigma_{f_1} & \nu\Sigma_{f_2} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}. \tag{3.1}$$

Solving the system in 3.1 for the infinite multiplication factor, the following analytic expression is obtained,

$$k = \frac{\Sigma_{a_2}\nu\Sigma_{f_1}+\Sigma_{1\to 2}\nu\Sigma_{f_2}}{\Sigma_{a_2}\left(\Sigma_{a_1}+\Sigma_{1\to 2}\right)}. \tag{3.2}$$

The infinite multiplication factor is a function of five material parameters. Since this thesis

is concerned with the affect of uncertainties in input parameters on computer code outputs, variations in k_∞ as a function its stochastic input variables are of interest. Assume all variation in k_∞ can be attributed to its input cross sections, whose distributions follow a multivariate Gaussian. To obtain physical homogenized, two-group cross section values a real system must first be modeled in a transport code. The Uncertainty Analysis in Modeling (UAM) Benchmark is sought for this purpose [20]. Specifically, the TMI assembly is modeled using the two-step method described in [40]. A total of 300 perturbed cross section sets were produced to obtain the few-group mean and covariance data used in the proceeding analysis. The cross section data is summarized in Table 3.1.

Table 3.1: Two-group cross section data for an infinite TMI lattice.

			Correlation Coefficient Matrix				
	Mean	**Standard Dev.**	Σ_{a_1}	Σ_{a_2}	$\nu\Sigma_{f_1}$	$\nu\Sigma_{f_2}$	$\Sigma_{1\to2}$
Σ_{a_1}	1.04E-02	9.06E-05	1	0.07	-0.13	0.02	0.75
Σ_{a_2}	1.10E-01	2.31E-04	0.07	1	0.06	0.31	-0.07
$\nu\Sigma_{f_1}$	9.00E-03	4.85E-05	-0.13	0.06	1	0.33	-0.10
$\nu\Sigma_{f_2}$	1.91E-01	8.87E-04	0.02	0.31	0.33	1	0.01
$\Sigma_{1\to2}$	1.80E-02	2.18E-04	0.75	-0.07	-0.10	0.01	1

Multiple methods will be applied to obtain basic statistical and sensitivity data on the infinite multiplication factor. Of course, Monte Carlo sampling using the input cross sections' covariance matrix and applying 2.40 will provide the mean and variance of k_∞. Another approach to get at the variance of k_∞ is through the "Sandwich Equation" [22],

$$\sigma^2(k_\infty) = S^T C S \tag{3.3}$$

where C is the covariance matrix for the input data. In equation 3.3 the array S contains sensitivities of the output to the input parameters. For this problem the vector S contains,

$$S^T = \left(\frac{\partial k_\infty}{\partial \Sigma_{a_1}} \quad \frac{\partial k_\infty}{\partial \Sigma_{a_2}} \quad \frac{\partial k_\infty}{\partial \nu\Sigma_{f_1}} \quad \frac{\partial k_\infty}{\partial \nu\Sigma_{f_2}} \quad \frac{\partial k_\infty}{\partial \Sigma_{1\to2}} \right). \tag{3.4}$$

Since there exists an analytic expression for k_∞ the sensitivity vector for this problem in 3.4 is exact. As a side-check the sensitivity vector S can also be constructed using central differencing. The sensitivity of k_∞ to the i^{th} cross section Σ_i using central differencing is expressed as,

$$\left.\frac{\partial k_\infty}{\partial \Sigma_i}\right|_{\Sigma_{j\neq i}=\bar{\Sigma}_j} \approx \frac{k_\infty(\Sigma_i + \Delta\Sigma_i) - k_\infty(\Sigma_i - \Delta\Sigma_i)}{2\Delta\Sigma_i} \tag{3.5}$$

where all cross sections Σ_j, $j \neq i$, are held at their mean values. Generally a one percent

perturbation $\Delta\Sigma$ is sufficient to obtain accurate sensitivities although this rule of thumb is dependent on the smoothness of the objective function. With a variety of methods available to obtain sensitivities and statistical momements of k_∞ it is possible to thoroughly assess the potential of a reduced order model.

3.1.2 Analysis

Several elements of the methodologies described in Chapter 2 will be tested in this section and compared to results obtained using analytic and Monte Carlo approaches. For all Smolyak sparse grids constructed the hypercube domain extends to six standard deviations in each random variable. Since k_∞ is a function of only five random variables a sparse grid interpolant will be constructed without applying any function decomposition in order to demonstrate the accuracy and convergence of the method. The convergence criteria for the sparse grid interpolant is set such that the maximum hierarchical surplus at a given level is not to exceed 10^{-10}. Both Clenshaw-Curtis and Gauss-Patterson abscissas are tested.

From Fig. 3.1 the Clenshaw-Curtis and Gauss-Patterson schemes perform similarly in terms of level to level convergence. However, observe that at each interpolation level the Gauss-Patterson scheme requires significantly more nodes in exchange for a small increase in convergence speed. Both schemes converge to the threshold around level five although the Gauss-Patterson scheme requires more than twice as many function evaluations to get there than Clenshaw-Curtis. Based on the graphical determination of order of convergence [7], it is clear from 3.1 that the Smolyak interpolant for k_∞ converges geometrically.

With the Smolyak interpolation routines working as expected it's safe to apply them to an anchored-ANOVA decomposition of k_∞. To start, only the first order components will be built and analyzed. The first order components are relatively cheap to produce and often collectively produce very accurate reduced order models [29]. Afterwards, all higher order components will be added in order to show that the full anchored-ANOVA decomposition can fully reproduce the objective function. Since k_∞ is a function of only five random variables there is no point in adaptively constructing the reduced order model as described in Section 2.3.3.

As a first comparison between all the models developed for quantifying the uncertainty in k_∞ the mean and variance values of each model will be compared. With the exception of the variance obtained using the Sandwich Equation, each model's variance was obtained by propagating 1000 samples of Eq. 2.40 through the model. All samples produced for each model were seeded identically and so the same random numbers were drawn. The mean and variance results, along with 99% confidence intervals, are summarized in Table 3.2.

Figure 3.1: Convergence study of a five dimensional sparse grid interpolant for the multiplcation factor of an infinite TMI lattice. The boxed numbers represent the current number of knots in the sparse grid.

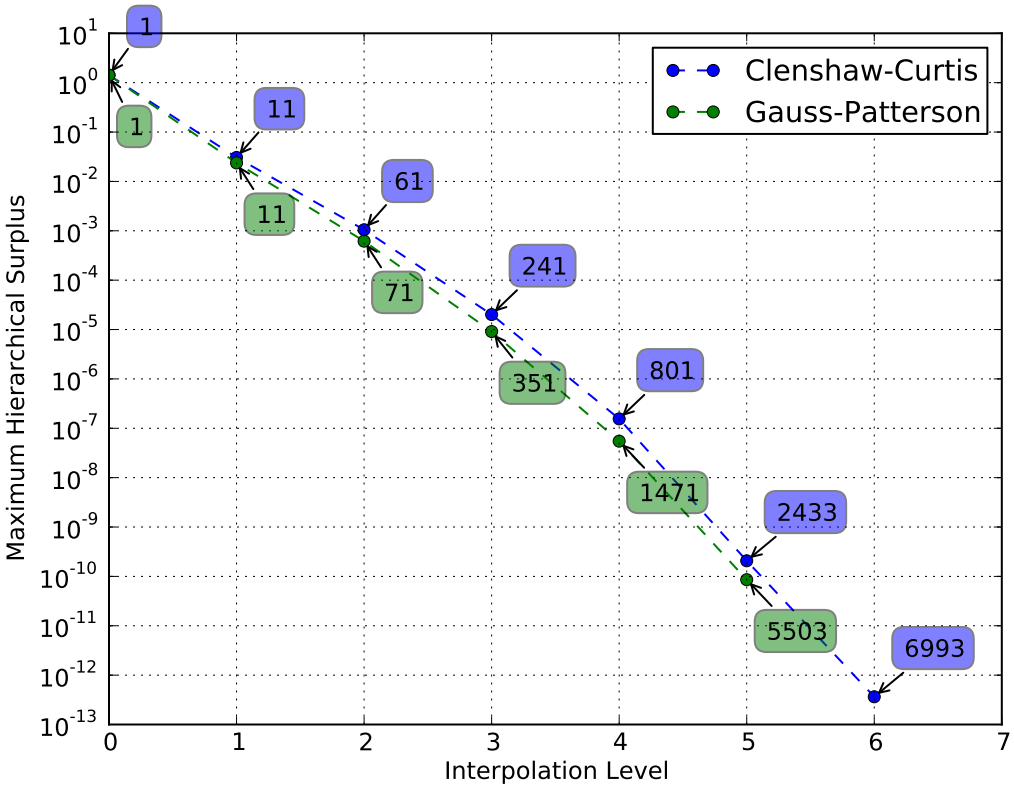
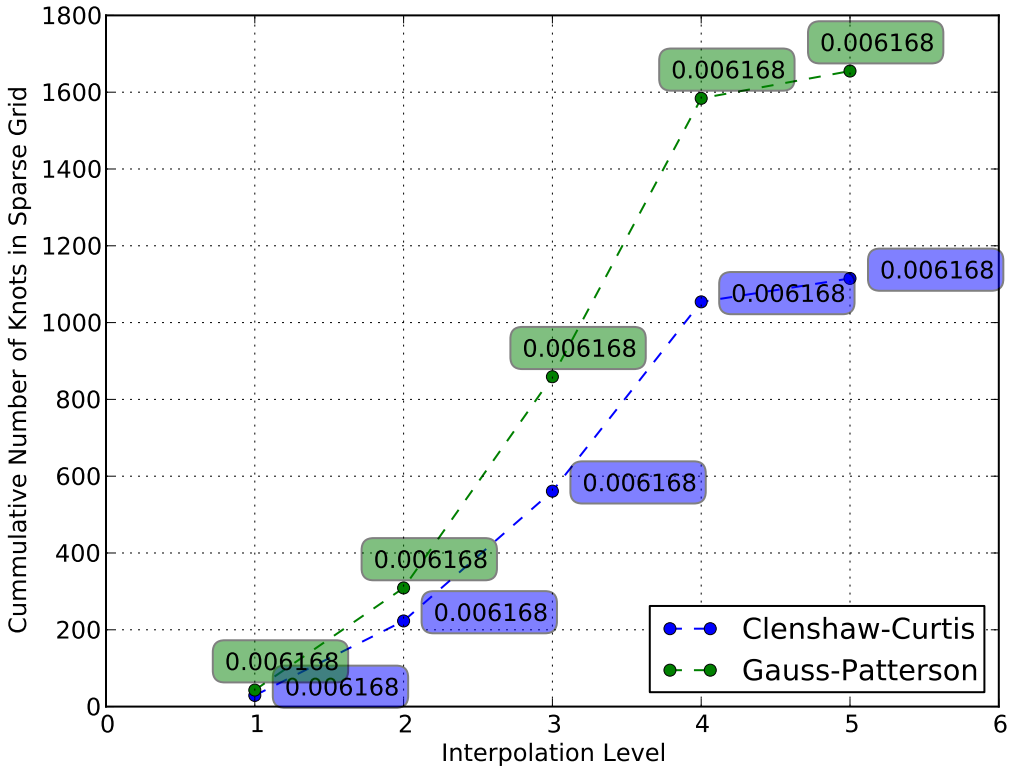


Table 3.2: Mean and variance data for the multiplication factor of an infinite TMI lattice obtained using Monte Carlo sampling. Wherever sampling was utilized the same random numbers were used.

Method	**Mean**	**99% CI**	**Standard Dev.**	**99% CI**
5D Sparse Grid CC	1.41562	(1.41512, 1.41612)	0.006168	(0.005909, 0.006544)
5D Sparse Grid GP	1.41562	(1.41512, 1.41612)	0.006168	(0.005831, 0.006544)
1D ANOVA CC	1.41560	(1.41510, 1.41610)	0.006168	(0.005831, 0.006544)
All ANOVA CC	1.41562	(1.41512, 1.41612)	0.006168	(0.005831, 0.006544)
1D ANOVA GP	1.41560	(1.41510, 1.41610)	0.006168	(0.005831, 0.006544)
All ANOVA GP	1.41562	(1.41512, 1.41612)	0.006168	(0.005831, 0.006544)
True Function	1.41562	(1.41512, 1.41612)	0.006168	(0.005831, 0.006544)
Sandwich			0.006540	

Figure 3.2: Cummulative number of knots required at each level of an anchored-ANOVA decomposition of the multiplication factor of an infinite TMI lattice. Boxes contain the calculated standard deviation at the current level.



The five dimensional sparse grid interpolant results are entirely self consistent with the anchored-ANOVA results. Further, both of these methods produce identical results to those obtained using Monte Carlo sampling. Although the analytic variance from the Sandwich Equation is within the 99% confidence bounds of each model's results, there is a notable difference due to the fact that only 1000 samples were used to obtain each model's statistics. Increasing the number of samples decreased the difference. Note that in Table 3.2 the anchored-ANOVA the reduced order models consisting of only one dimension anchored-ANOVA components perfrom just as well as the full decomposition and the sparse grid interpolants over all five random variables. However, the 1D component models require only 29 function evaluations to produce, which is some ten times fewer evaluations than the 5D interpolants, and some hundred times fewer evaluations than the full decomposition. The rapid convergence of the reduced order model containing only one dimensional anchored-ANOVA components is shown in 3.2. Construction of higher order components

Table 3.3: Normalized sensitivity coefficients for the multiplication factor of an infinite TMI lattice.

	Normalized Sensitivity Coefficient of k_∞				
Method	Σ_{a_1}	Σ_{a_2}	$\nu\Sigma_{f_1}$	$\nu\Sigma_{f_2}$	$\Sigma_{1\to 2}$
5D Sparse Grid CC	-.367551	-.776087	.224060	.776010	.143491
5D Sparse Grid GP	-.367551	-.776087	.224060	.776010	.143491
1D ANOVA CC	-.367556	-.776098	.224063	.776020	.143493
All ANOVA CC	-.367551	-.776087	.224060	.776010	.143491
1D ANOVA GP	-.367556	-.776098	.224063	.776020	.143493
All ANOVA GP	-.367551	-.776087	.224060	.776010	.143491
Analytic	-.367520	-.775956	.224044	.775956	.143476
Central Difference	-.367551	-.776089	.224060	.776011	.143492

is very expensive. Fortunately for this problem, and perhaps others, construction of only one dimensional components is completely sufficient to represent the objective function.

As another performance measure of the reduced order model methodologies, each model is used to obtain normalized sensitivity coefficients for k_∞. Central differencing is applied to each model, with perturbations made to each cross section at a time while holding the other cross sections at their mean values. Perturbations are taken to be 1% of each cross section's value. Using the analytic expression for k_∞ in 3.2, the central differencing results can be compared to the true sensitivity coefficients. The results are summarized in Table 3.3. Table 3.3, sensitivity coefficients are also obtained by applying central differencing to the true function as in Eq. 3.5. As expected, all models utilizing the central differencing formula produce self consistent sensitivity coefficients. The models differ from the analytic sensitivity coefficients only in the fourth decimal place, which is expected given the $\mathcal{O}(\Delta\Sigma^2)$ convergence of the central differencing formula.

3.2 Point Kinetics/Lumped Thermal Hydraulics

3.2.1 Problem Statement

A reduced order model based on the anchored-ANOVA decomposition will be constructed in this section for a simple system of ordinary differential equations modeling a transient in a BN800 sodium fast cooled reactor. The physical model of the reactor consists of point kinetics to model the neutronics and lumped thermal hydraulics equations to describe temperature feedback. The coupled system is nonlinear and only has a time dependence. Previous research groups have utilized point kinetics and lumped thermal hydraulics equa-

tions to model basic reactor systems in [15], [14], and [18]. In this section a reduced order model will be constructed for the maximum fuel temperature attained following a reactivity insertion as a function of the random variables exhibited in the description of the point kinetics/lumped thermal hydraulics system.

The six-group point kinetics equations modeling the neutronics of a reactor consist of a balance for reactor power $P(t)$ and a balance equation for each of the six precursor concentrations $C_k(t)$. Changes in reactor power are dependent on the precursor concentration, decays constants λ_k, delayed neutron fraction β and the mean neutron generation time Λ as detailed in,

$$\frac{dP}{dt} = \frac{\rho(T_f, T_c, t) - \beta}{\Lambda} P + \sum_{k=1}^{6} \lambda_k C_k. \tag{3.6}$$

The reactivity ρ depends on feedback from the fluids temperature models for the reactor fuel and coolant, which in turn depend on reactor power. The expression for each of the k precusor concentrations is written as,

$$\frac{dC_k}{dt} = -\lambda_k C_k + \frac{\beta_k}{\Lambda} P. \tag{3.7}$$

As for the ordinary differential equations describing the behavior of the reactor coolant system, two coupled equations suffice. For the fuel temperature T_f, the following lumped model is used,

$$M_f c_{pf} \frac{dT_f}{dt} = P + Ah(T_c - T_f) \tag{3.8}$$

where M_f is the lump fuel mass, c_{pf} is the specific heat capacity of the fuel, A is the heat transfer surface, and h is the heat transfer coefficient between the coolant and reactor fuel. Finally, the coolant temperature is described as,

$$M_c c_{pc} \left(\frac{dT_c}{dt} + v \frac{T_c - T_{in}}{L} \right) = Ah(T_f - T_c) \tag{3.9}$$

where M_c is the lump coolant mass, c_{pc} is the specific heat capacity of the coolant, L is the coolant channel length, v is the coolant flow velocity, and T_{in} is the inlet coolant temperature. The initial conditions for P, C_k, T_f, and T_c depend on the initial power in the

reactor P_0 before any kind of transient occurs and are listed in 3.10.

$$\begin{aligned} P(0) &= P_0 \\ C_k(0) &= \frac{\beta_k}{\lambda_k \Lambda} P_0 \\ T_f(0) &= T_c(0) + \frac{P_0}{Ah} \\ T_c(0) &= T_{in} + \frac{P_0 L}{M_c c_{pc} v} \end{aligned} \tag{3.10}$$

Serving as the coupling device between the lumped thermal hydraulics model and point kinetics model is the reactivity, which is proportional to the coolant temperature and the fuel temperature. Of course, any external reactivity ρ_{ex} added to the reactor is also a contributor. The time dependent reactivity is given explicitly as,

$$\rho(T_f, T_c, t) = \rho_{ex} + \alpha_d(T_f - T_f(0)) + \alpha_c(T_c - T_c(0)) \tag{3.11}$$

where α_d and α_c are the doppler and coolant coefficients of reactivity, respectively.

The equations in 3.6, 3.7, 3.9, and 3.8 are used to model the transient resulting from a half sawtooth external reactivity insertion, as shown in 3.12.

$$\rho_{ex}(t) = \begin{cases} t\rho_{max}/20 & t \leq 20 \\ 0 & t > 20 \end{cases} \tag{3.12}$$

By treating the coefficeints in the point kinetics/lumped thermal hydraulics model as random variables, the objective function investigates the response surface for the maximum fuel temperature attained during transient. The reduced order methodologies will be tested against the stated problem. A depiction of the transient at the random variables' mean values, along with the external reactivity is shown in Figure 3.3. A total of twenty two random variables will be investigated for their affect on the maximum fuel temperature attained during transient. The random variables' mean values, along with their standard deviations are listed in 3.4. Note that all standard deviations are taken to be 5% of the mean value. All random variables are assumed to be independent of one another, as was assumed in [14]. A plot of the fuel temperature as a function of time due to the external reactivity profile shown in the same figure is depicted in Figure 3.4.

Figure 3.3: Transient resulting from a half sawtooth external reactivity insertion, as modeled using the mean parameter values of the point kinetics/lumped thermal hydraulics system.

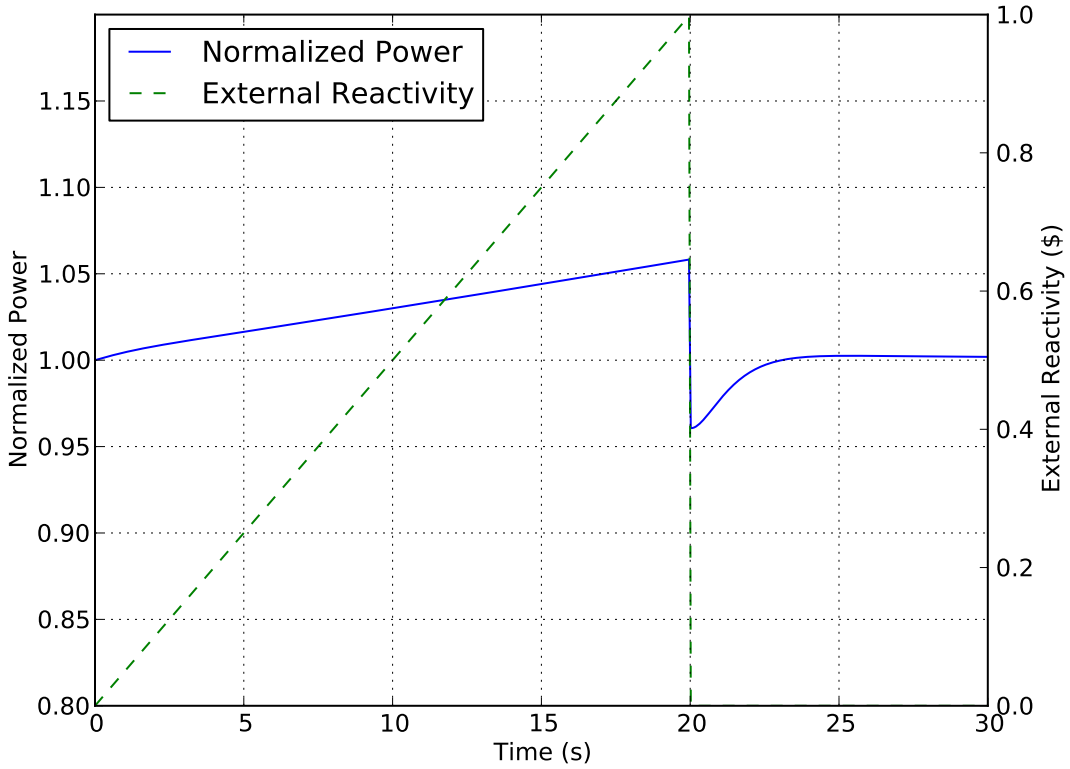
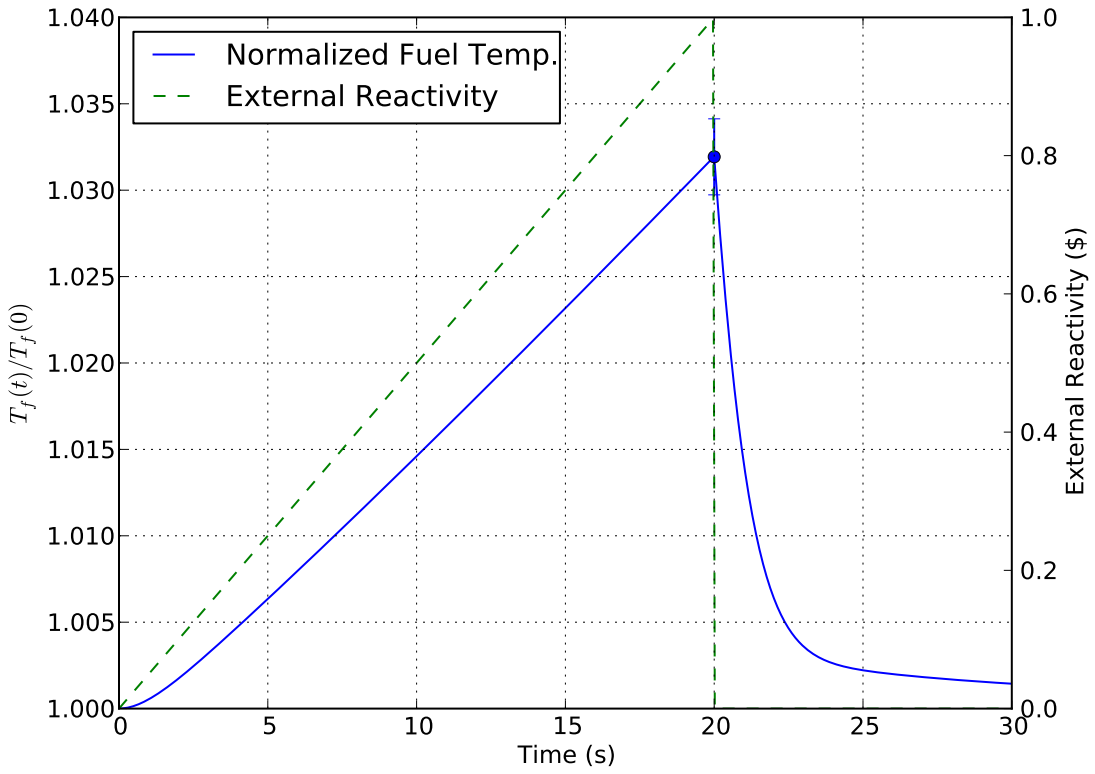


Table 3.4: Mean parameter values used in the point kinetics/lumped thermal hydraulics model for the analysis of a BN800 fast sodium cooled reactor.

Random Variable	**Units**	**Mean**	**Standard Dev.**
λ_1	s^{-1}	1.24E-02	6.20e-04
λ_2	s^{-1}	3.05E-02	1.52e-03
λ_3	s^{-1}	1.11E-01	5.55e-03
λ_4	s^{-1}	3.01E-01	1.50e-02
λ_5	s^{-1}	1.14E+00	5.70e-02
λ_6	s^{-1}	3.01E+00	1.50e-01
β_1		9.00E-05	4.50e-06
β_2		8.53E-04	4.26e-05
β_3		7.00E-04	3.50e-05
β_4		1.40E-03	7.00e-05
β_5		6.00E-04	3.00e-05
β_6		5.50E-04	2.75e-05
Λ	s	4.00E-07	2.00e-08
Ah	kW/K	2.50E+06	1.25e+05
M_c	kg	1.16E+03	5.84e+01
M_f	kg	9.67E+03	4.83e+02
c_{pc}	$J/kg \cdot K$	1.20E+03	6.00e+01
c_{pf}	$J/kg \cdot K$	5.00E+02	2.50e+01
v	m/s	7.50E+00	3.75e-01
α_d	pcm/K	6.87E-06	3.43e-07
α_c	pcm/K	1.23E-06	6.15e-08
ρ_{max}		4.19E-04	2.09e-05

Figure 3.4: Fuel temperature transient resulting from a half sawtooth reactivity insertion. All parameters in the coupled point kinetics/lumped thermal hydraulics equations are held at their mean values.



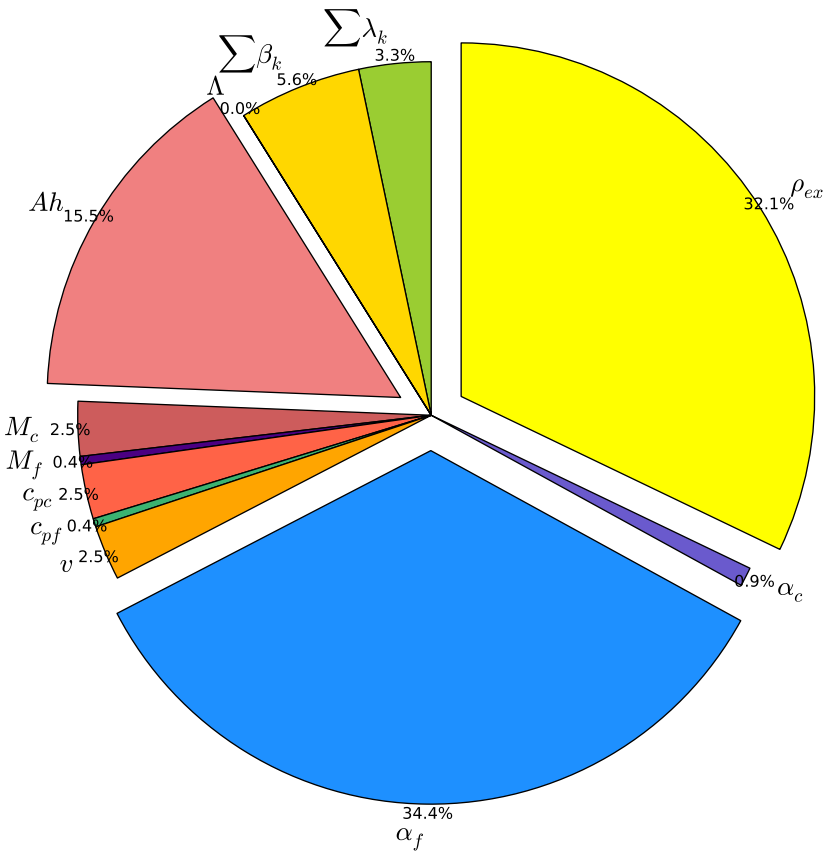
3.2.2 Analysis

An adaptive reduced order model, whose formulation is summarized in Algorithm 4, will be created for the problem described in section 3.2.1. The reduced order model will be investigated for its ability to reproduce statistics of interest by comparing its results with those obtained from sampling the true function. As described in Algorithm 4, the first step in creating a reduced order model for the maximum fuel temperature is to construct all first order components in the anchored-ANOVA decomposition and to identify the important ones. The sparse grids comprising the reduced order model are assumed to be converged when the maximum hierarchical surplus for a given level is less than 10^{-5}. Consequently, at least five digits of accuracy are expected. Important dimensions are those whose normalized sensitivity index in Eq. 2.43 exceeds 5%.

From Figure 3.5 the "important" variables are identified to be Ah, α_d, and ρ_{max}. Collec-

tively these three "important" random variables comprise 82% of the total sensitivity. The

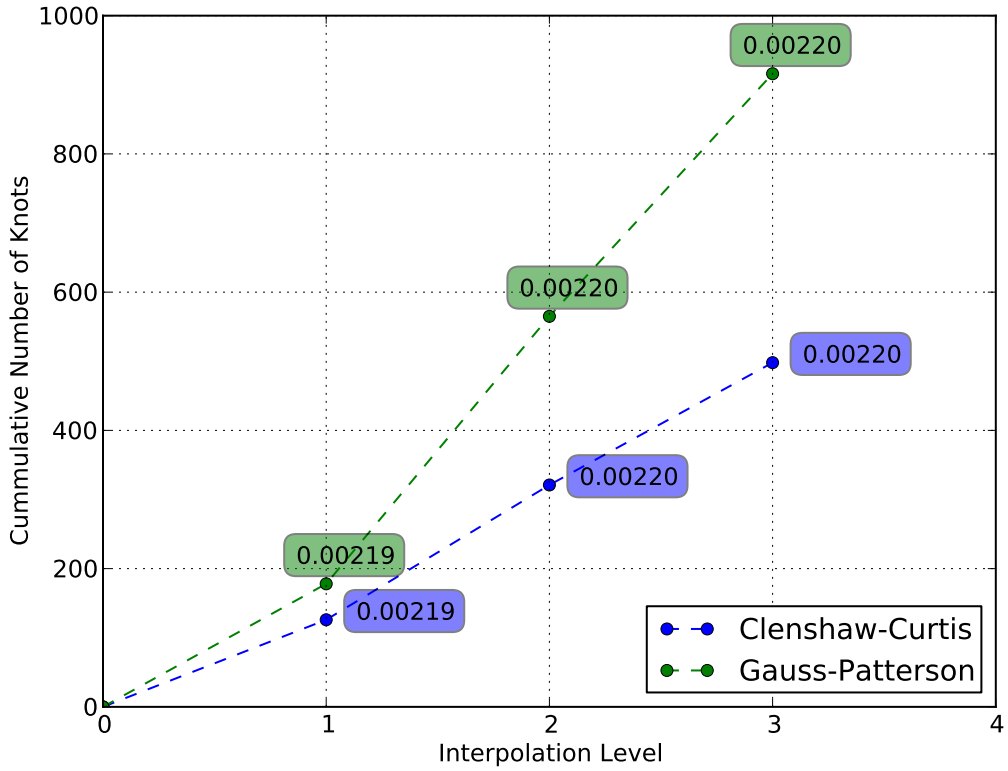
Figure 3.5: Normalized sensitivity indices for all random variables comprising the coupled point kinetics/lumped thermal hydraulics equations. The effects of all β_k and λ_k have been lumped into single β a λ effects, respectively.



indication that the maximum fuel temperature is sensitive to the heat transfer Ah from fuel to coolant is not surprising since in Eq. 3.8 the fuel temperature is directly proportional to Ah. Further, the random variables ρ_{max} and α_d determine the slope of the increase in fuel temperature, as seen in Figure 3.4, and so a strong sensitivity to these random variables is expected. The sensitivity of the maximum fuel temperature to α_c is not as great since the increase in coolant temperature during the transient is significantly smaller than the rise in fuel temperature. Relatively weak sensitivity to M_f and c_{pf} can perhaps be attributed to cancellation of error since these two variables are multiplied together.

With only three random variables deemed as "important" only three second order anchored-ANOVA components must be built for the reduced order model. Neglecting any convergence critiera, third order component describing the interaction effects among the three important random variables is also built. A summary of the total number of function evaluations needed to construct the adaptive reduced order model for the maximum fuel temperature is shown in Figure 3.6. The Gauss-Patterson scheme required almost twice as many

Figure 3.6: Number of knots needed to adaptively construct a reduced order model for the maximum fuel temperature in both the Clenshaw-Curtis and Gauss-Patterson schemes. Boxed values state the standard deviation calculated at each level.



knots as Clenshaw-Curtis to adaptively build the reduced order model. From Figure 3.6 it's clear that the reduced order model consisting of only one dimensional components is effectively just as accurate as the full model, but requiring only 126 function evaluations to build using Clenshaw-Curtis. To see how well the reduced order models are able to reproduce the mean and variance of the true function Monte Carlo simulation is utilized. The models produced using anchored-ANOVA decomposition with superposition dimensions of one and three are sampled along with the true function. Mean, variance, and pertinent

99% confidence intervals for the sampling are summarized in Table 3.5. A total of 1000 samples were used for each method, each using the same random numbers. As evidenced

Table 3.5: Mean and variance data for the maximum fuel temperature achieved during transient obtained using Monte Carlo sampling. The same random numbers were used for all 1000 samples for each method.

Method	**Mean**	**99% CI**	**Standard Dev.**	**99% CI**
1D ANOVA CC	1.03193	(1.03175, 1.03211)	0.002187	(0.002068, 0.002320)
All ANOVA CC	1.03193	(1.03175, 1.03211)	0.002196	(0.002076, 0.002330)
1D ANOVA GP	1.03193	(1.03175, 1.03211)	0.002187	(0.002068, 0.002320)
All ANOVA GP	1.03193	(1.03175, 1.03211)	0.002196	(0.002076, 0.002330)
True Function	1.03193	(1.03175, 1.03211)	0.002196	(0.002076, 0.002330)

in Table 3.5 the statistical results for each method are consistent. While the reduced order model with superposition dimension of three is able to replicate the Monte Carlo results to five significant figures, the expected accuracy, the order-one superposition model is slightly short. Of course, this is due to the absence of higher order components. However, the proximity of the order-one superposition model's results to the true results indicate that for this problem the higher order components do not have a significant impact. To further verify the ability of the reduced order models to accurately reproduce basic statistical moments, the probability distributions for the normalized maximum fuel temperature produced by each model are compared in Figure 3.7. All of the tested reduced order models are able to reproduce the Gaussian probability distribution for the normalized maximum fuel temperature.

As done in section 3.1.2, a sensitivity analysis will be completed for the reduced order model and compared to the normalized sensitivity coefficients obtained using central differencing. From Table 3.6 notice that the largest sensitivity coefficients are those of the random variables deemed "important" using the adaptive reduced order model algorithm. Only the sensitivity coefficients for the Clenshaw-Curtis sparse grid are shown in Table 3.6 since the Gauss-Patterson sparse grid returns identical results. The normalized sensitivity coefficients calculated using the reduced order models are the same as those calculated using central differencing to the expected number of significant digits.

3.3 TMI Minicore

3.3.1 Problem Statement

The previous two problems dealt with relatively simple functions that don't require industrial engineering codes to solve. However, the main intention of this thesis is to construct

Figure 3.7: Histograms produced by sampling the true function, order-one superposition reduced order model, and the full adaptive reduced order model.

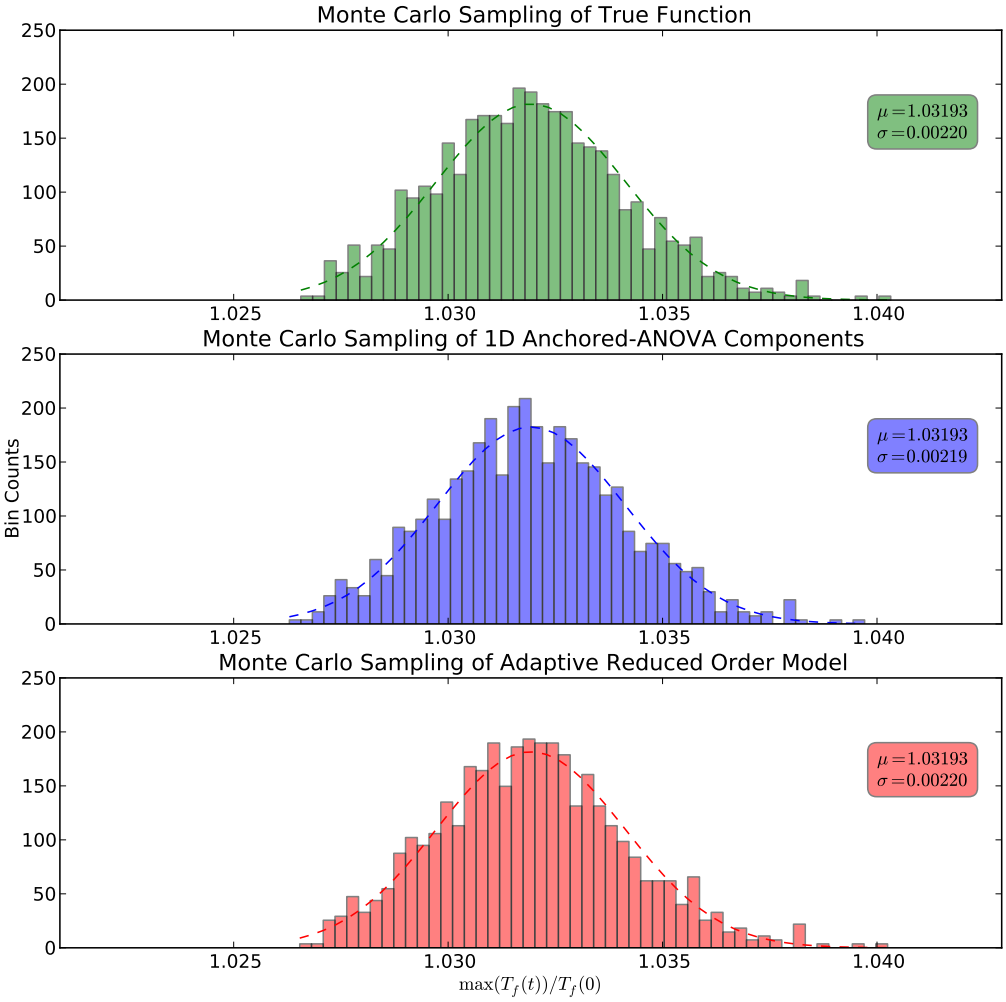


Table 3.6: Normamlized sensitivity coefficients of the maximum fuel temperature to random variables.

Random Variable	**1D ANOVA CC**	**All ANOVA CC**	**Central Diff.**
λ_1	3.7894E-05	3.7894E-05	3.7895E-05
λ_2	7.0387E-04	7.0387E-04	7.0387E-04
λ_3	8.4244E-04	8.4244E-04	8.4215E-04
λ_4	9.7308E-04	9.7309E-04	9.7379E-04
λ_5	1.1572E-04	1.1572E-04	1.1607E-04
λ_6	4.4638E-05	4.4639E-05	4.0498E-05
β_1	-3.2992E-04	-3.2992E-04	-3.2976E-04
β_2	-2.6582E-03	-2.6582E-03	-2.6616E-03
β_3	-1.1953E-03	-1.1953E-03	-1.2040E-03
β_4	-1.0129E-03	-1.0129E-03	-1.0232E-03
β_5	-1.1689E-04	-1.1689E-04	-1.1810E-04
β_6	-4.0718E-05	-4.0718E-05	-4.1134E-05
Λ	-8.9294E-08	-8.9295E-08	-8.9364E-08
Ah	1.2553E-02	1.2553E-02	1.2584E-02
M_c	1.8753E-03	1.8753E-03	1.8716E-03
M_f	-3.6695E-04	-3.6695E-04	-3.6360E-04
c_{pc}	1.8753E-03	1.8753E-03	1.8903E-03
c_{pf}	-3.6695E-04	-3.6695E-04	-3.5976E-04
v	1.8838E-03	1.8839E-03	1.9177E-03
α_d	-2.6655E-02	-2.6656E-02	-2.6625E-02
α_c	8.4387E-04	8.4387E-04	8.7194E-04
ρ_{max}	3.1164E-02	3.1164E-02	3.1272E-02

reduced order models for computer codes that aim to model large engineering systems. Interaction with such computer codes consist of input and output files; the governing equations and their solvers are rarely seen. The primary purpose of this demonstration problem is to show that the same algorithms applied to analyze the previous problems are also functional when applied to engineering computer codes.

In this demonstration problem the reactor core simulator code Purdue Advanced Reactor Core Simulator (PARCS) [10] is applied to the TMI minicore described in the first phase of the UAM Benchmark [20].

BIBLIOGRAPHY

[1] H.S. Abdel-Khalik and P.J. Turinsky. Evaluation of core attributes uncertainties due to input data uncertainties. volume 92 of *Transactions of the American Nuclear Society*, San Diego, CA, 2005.

[2] Nitin Agarwal and N.R. Aluru. A domain adaptive stochastic collocation approach for analysis of mems under uncertainties. *Journal of Computational Physics*, 228:7662–7688, 2009.

[3] John K. Au. *An Ab Initio Approach to the Inverse Problem-based Design of Photonic Bandgap Devices*. PhD thesis, California Institute of Technology, 2007.

[4] Volker Barthelmann, Erich Novak, and Klaus Ritter. High dimensional polynomial interpolation on sparse grids. *Advances in Computational Mathematics*, 12:273–288, 2000.

[5] Jean-Paul Berrut and Lloyd N. Trefethen. Barycentric lagrange interpolation. *SIAM Review*, 46(3):501–517, 2004.

[6] S.M. Bowman. Scale 6: Comprehensive nuclear safety analysis code system. *Nuclear Technology*, 174(2):126–148, 2011.

[7] John Boyd. *Chebyshev and Fourier Spectral Methods*. Dover Publications, 2 edition, 2001.

[8] John Burkardt. Sparse grid collocation for uncertainty quantification. SCALA, Louisiana State University, 2012.

[9] Philip Davis and Philip Rabinowitz. *Methods of Numerical Integration*. Dover, 2007.

[10] Thomas Downar. Parcs: Purdue advanced reactor core simulator. Proceedings of the International Conference on the New Frontiers of Nuclear Technology, Seoul, South Korea, 2002.

[11] James Duderstadt and Louis Hamilton. *Nuclear Reactor Analysis*. John Wiley and Sons, 1976.

[12] Zhen Gao and Jan S. Hesthaven. On anova expansions and strategies for choosing the anchor point. *Applied Mathematics and Computation*, 217:3274–3285, 2010.

[13] Zhen Gao and Jan S. Hesthaven. Efficient solution of orindary differential equations with high-dimensional parameterized uncertainty. *Communications in Computational Physics*, 10(2):253–278, 2011.

[14] L. Gilli, D. Lathouwers, J.L. Kloosterman, and T.H. van der Hagen. Application of sensitivity analysis to a simplified coupled neutronic thermal-hydraulics transient in a fast reactor using adjoint techniques. In *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Rio de Janiero, Brazil, May 2011. American Nuclear Society.

[15] L. Gilli, D. Lathouwers, J.L. Kloosterman, and T.H. van der Hagen. Performing uncertainty analysis of a nonlinear point-kinetics/lumped paremeters problem using polynomial chaos techniques. *Annals of Nuclear Energy*, 40:35–44, 2012.

[16] Peter Henrici. *Essentials of Numerical Analysis with Pocket Calculator Demonstrations*. John Wiley and Sons, 1982.

[17] Markus Holtz. *Sparse Grid Quadrature in High Dimensions with Applications in Finance and Insurnance*. Springer, 2011.

[18] C. Housiadas. Lumped paramaters analysis of coupled kinetics and thermal-hydraulics for small reactors. *Annals of Nuclear Energy*, 29:1315–1325, 2002.

[19] Gianluca Iaccarino. Uncertainty quantification in computational science. Lehrstuhl fr Aerodynamik und Strmungsmechanik, Technische Universitt Mnchen, 2011.

[20] K. Ivanov, M. Avramova, I. Kodeli, and E. Sartori. Benchmark for uncertainty analysis in modeling for design, operation and safety analysis of lwrs. Technical report, Nuclear Energy Agency, 2007.

[21] M.A. Jessee, H.S. Khalik, and P.J. Turinsky. Evaluation of bwr core attributes uncertainties due to multi- group cross section uncertainties. In *Joint International Topical Meeting on Mathematics and Computation and Supercomputing in Nuclear Applications*, Monterey, CA, April 2007. American Nuclear Society.

[22] M.A. Jessee, P.J. Turinsky, and H.S. Abdel-Khalik. Many-group cross-section adjustment techniques for boiling water reactor adaptive simulation. *Nuclear Science and Engineering*, 169:40–55, 2011.

[23] H.F. Kaiser and K. Dichman. Sample and population score matrices and sample correlation matrices from an arbitrary population correlation matrix. *Psychometrica*, 27(2):179–182, 1962.

[24] M. Klein, L.Gallner, B. Krzykacz-Hausmann, I. Pasichnyk, A. Pautz, and W. Zwermann. Influence of nuclear data covariance on reactor core calculations. In *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Rio de Janeiro, Brazil, May 2011. American Nuclear Society.

[25] Andreas Klimke and Barbara Wohlmuth. Peicewise multilinear hierarchical sparse grid interpolation in matlab. *ACM Transactions on Mathematical Software*, 31(4):561–579, 2005.

[26] Genyuan Li, Sheng-Wei Wang, and Herschel Rabitz. High dimensional model representations: Concepts and applications. *Journal of Physical Chemistry A*, 105:7765–7777, 2001.

[27] Meilin Liu, Zhen Gao, and Jan Hesthaven. Adaptive sparse grid algorithms with applications to electromagnetic scattering under uncertainty. *Applied Numerical Mathematics*, 61:24–37, 2011.

[28] Xiang Ma and Nicholas Zabaras. An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations. *Journal of Computational Physics*, 228:3084–3113, 2009.

[29] Xiang Ma and Nicholas Zabaras. An adaptive high-dimensional stochastic model representation technique for the solution of stochastic partial differential equations. *Journal of Computational Physics*, 229:3884–3915, 2009.

[30] O.P Le Maitre and Omar Knio. *Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics*. Springer, 2010.

[31] Brent Oksendal. *Stochastic Differential Equations: An Introduction with Applications*. Springer-Verlag, 6 edition, 2005.

[32] H.E. Salzer. Lagrangian interpolation at the chebyshev points $x_{n,v} = \cos(v\pi/n)$. *The Computer Journal*, 15(2):156–159, 1972.

[33] Sergei A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl.*, 4:240–243, 1963.

[34] Lloyd Trefethen. Is gaussian quadrature better than clenshaw-curtis? *SIAM Review*, 50(1):67–87, 2008.

[35] Lloyd Trefethen. *Approximation Theory and Approximation Practice*. SIAM, 2012.

[36] M.L. Williams. Perturbation theory for reactor analysis. In *CRC Handbook of Nuclear Reactor Calculations*, pages 63–188. CRC Press, 1986.

[37] M.L. Williams, J.C. Gehin, and W.S. Yang. Combining mc^2-2 and scale cross section methods to process nuclear data for gnep. In *International Conference on Reactor Physics, Nuclear Power: A Sustainable Resource*, Interlaken, Switzerland, September 2008.

[38] M.L. Williams, G. Ilas, M.A. Jessee, B.T. Rearden, D. Wiarda, W. Zwermann, L. Gallner, M. Klein, B. Krzykacz-Hausmann, and A. Pautz. A statistical sampling method for uncertainty analysis with scale and xsusa. *Nuclear Technology*, 183(3):515–526, 2012.

[39] Dongbin Xiu and Jan S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal of Scientific Computing*, 27:1118–1139, 2005.

[40] Artem Yankov, Benjamin Collins, and Markus Klein. A two-step approach to uncertainty quantification of core simulators. *Science and Technology of Nuclear Installations*, 2012.