ABSTRACT

SOGA, SHOTA. Subspace Methods applied to gPC-based Surrogate Model Construction. (Under the direction of Hany Abdel-Khalik.)

This thesis employs a recently developed reduced order modeling technique to render a more efficient use of polynomial chaos method for surrogate model construction. Our interest is in models containing many input parameters, which are often encountered in nuclear engineering applications. The existing generalized Polynomial Chaos(gPC) approach suffers from the curse of dimensionality when applied to models with many parameters. In this thesis, we hybridize recently developed reduced order modeling techniques based on the proper orthogonal decomposition method with gPC to reduce the computational cost required to build the surrogate. The goal is to use ROM techniques to reduce the effective dimension of the model either at the state space or the input parameters space. With the dimension of the model reduced, the computational cost required for gPC can be substantially reduced. We also explore the use of sparse approximation methods to find only the most dominant components of the gPC therefore further reducing the computational cost. To study the impact of the various reduction techniques, several hybridization approaches are employed. All developments are demonstrated using a neutronics model that calculates the neutron distribution inside a reactor. Two types of reduction are employed, a reduction at the state space and a reduction at the parameter space. In the state space, we use the traditional POD approaches to find a subspace of the state space that captures the most dominant variations for the state. At the parameters space, we use a recently developed approach that employs gradient information to find a subspace in the parameter space, referred to as the active subspace. The inactive subspace, the orthogonal complement to the active subspace, has much higher dimension than the active subspace, implying that the model depends only on a few degrees of freedom. This property is used to recast the gPC expansion by constraining it to the active subspace.

Subspace Methods applied to gPC-based Surrogate Model Construction

by
Shota Soga

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Nuclear Engineering

Raleigh, North Carolina

2012

APPROVED BY:

Ralph C. Smith	Yousry Y. Azmy
Clayton G. Webster	Hany Abdel-Khalik Chair of Advisory Committee

DEDICATION

To my parents.

BIOGRAPHY

Shota Soga was born on August 15th, 1987 in Gifu, Japan. He attended University of Tennessee, Knoxville, TN from 2007 to 2011 where he obtained a Bachelor of Science degree in Nuclear Engineering with Summa Cum Laude. Immediately after graduation, he joined the Nuclear Engineering department at North Carolina State University in pursuit of his Master of Science degree.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this thesis. I would first like to thank my parents for their support. Without them, I would not have been fortunate to be where I am today. I would like to thank Dr. Abdel-Khalik for his invaluable advice, guidance, and mentorship. Dr. Webster has hosted me for few days visit at ORNL to learn about the basic implementation of sparse grid methods. Finally, I am thankful to my committee members for their thoughts and feedback.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background and Motivation

Nuclear reactor calculations are extremely complicated due to the complex nature of the physics of radiation interaction with matter as well as the heterogeneous nature of reactor design, which often contains a great deal of spatial details. The complexity of these models makes it difficult to execute the engineering-oriented analyses such as sensitivity and uncertainty analysis, essential to assess the credibility of the simulation. Any of these analyses require many executions of the model, often proportional to and exponentially dependent on the number of input parameters. For typical reactor models, the number of parameters can range from between hundreds to millions depending on the level of details captured by the model. This renders uncertainty analysis computationally intractable for reactor calculations. To overcome this problem, the scientific community has focused on using reduced order modeling techniques to replace the original complex code with an inexpensive surrogate model that can be executed repeatedly for various engineering analyses. There are a great number of methods developed over the years for surrogate model construction, each with its own advantages and disadvantages. A common theme to these methods is that the computational cost required to build the surrogate is also dependent on the number of parameters, which makes its construction difficult. Moreover, the accuracy of the surrogate is often difficult to quantify, which is a pre-requisite for ensuring the results of all subsequent analyses using the surrogate are credible. We focus in this thesis on a class of reduced order modeling techniques, referred to hereinafter as subspace methods. In subspace methods the original dimensions of the

model are reduced by confining variations to a subspace, called active subspace. The implication is that variations orthogonal to the active subspace are not significant and can be ignored. If the size of the active subspace is small enough, huge reduction in the computational cost for surrogate construction can be achieved. More importantly, using recent developments, one can show that the errors resulting from the reduction can be bounded to ensure credibility of the surrogate predictions.

Subspace Approach as Reduced Order Modeling

Historically, engineering applications of reduced order modeling techniques have gained popularity in the hydraulics community because of the high computational cost of the models under consideration, but reduced order modeling itself can be traced back to the mathematics of interpolation, truncated Taylor expansion, Method of Weighted Residuals and so on. Reduced order modeling are be defined as a process by which solutions to complex models can be described using few degrees of freedom, smaller than the dimension of the complex solution. For example, assuming the model solution is described by a function in one dimension evaluated at N points as determined by the discretization technique used to obtain the numerical solution. If one can approximate this function by a number of select functions, say r which is less than N, then one has a reduced order model. This follows as now the function depends on r degrees of freedom rather than N. One may argue that any engineering model can be considered a reduced order model of the exact model (often unknown). This follows since the physical phenomenon under consideration often has large degrees of freedom than engineering models. ROM techniques can be distinguished based on the type of functions employed for the expansion. Once an appropriate selection is made, one can re-cast the original model into one with much fewer equations, which can be readily solved more efficiently. The choice of the expansion functions is fundamental to the success of this reduction approach. This choice is expected to depend on the model, and therefore expert judgment is often needed to make an educated choice. Moreover, calculating the errors resulting from this reduction is an important step of the analysis. This error can be estimated a posteriori, i.e., after the reduced model is constructed, by comparing its predictions to the original models.
Important examples of such approaches are the spectral expansion method and the method of weighted residuals [2, 3]. In both methods, a set of orthogonal functions can be shown to result in a rate of convergence that is exponential with respect to the order

of the expansion when the solution is sufficiently smooth [2].

The most common engineering approaches for reduced order modeling are the Proper Orthogonal Decomposition Galerkin projection approach [4, 5]. The Proper Orthogonal Decomposition is one of the common subspace approaches found in engineering disciplines. In the subspace approach, the structure of the solution to the system of stochastic PDEs is represented by the linear combination of minimum number of the orthogonal basis functions or vectors. In this method, the original system of PDEs is projected onto the reduced order subspace generated by the proper orthogonal decomposition. This method is extensively applied in fluid dynamics community. One common example is a linear matrix equation of the form:

$$\mathbf{A}x = b \tag{1.1}$$

and some results are demonstrated [4, 5].

The k-Eigenvalue Problem in Neutronics

In neutronics problems, most problems can be categorized into two major matrix forms:
1) Source-driven problem:

$$\mathbf{L}\phi = q_{\text{source}}. \tag{1.2}$$

2) Generalized eigenvalue problem:

$$\mathbf{L}\phi = \frac{1}{k}\mathbf{F}\phi. \tag{1.3}$$

In the source-driven problem, Eq.(1.2), the structure of the problem is the same as Eq.(1.1) and the Proper Orthogonal Decomposition Galerkin projection approach can be easily applied. However, in the case of the generalized eigenvalue problem, the method is not directly applicable. Historically in nuclear engineering community, generalized eigenvalue problem has been solved by the power iterative method and its variants which are summarized in the following algorithm**??**. In this thesis we propose two approaches to extend the applicability of POD techniques to generalized eigenvalue problems, one relying on forward and the other on adjoint formulations.

Algorithm 1 Pseudo Algorithm for Power Iterative Method [6]

Set $k_0 = k_{initial}$

Set the initial guess, $\phi^0 = \phi^{initial}$

$s = 0$

while $||\phi^{(s)} - \phi^{(s-1)}|| > \epsilon_{\phi,1}(1 - \rho_\phi^{(s)})$ or $||k^{(s)} - k^{(s-1)}|| > \epsilon_k(1 - \rho_k^{(s)})$ **do**

$s = s + 1$

Calculate $q_{source}^{(s)} = \frac{1}{k^{(s-1)}}\mathbf{F}\phi^{(s-1)}$

Solve $\mathbf{L}\phi^{(s)} = q_{source}^{(s)}$.

Update k-eigenvalue, $k^{(s)} = k^{(s-1)}\frac{\int dV \nu_f \Sigma_f \phi^{(s)}}{\int_V dV \nu_f \Sigma_f \phi^{(s-1)}}$

Approximate the spectral radius of $\phi^{(s)}$, $\rho_\phi^{(s)} = \frac{||\phi^{(s)} - \phi^{(s-1)}||_\infty}{||\phi^{(s-1)} - \phi^{(s-2)}||_\infty}$

Approximate the spectral radius of $k^{(s)}$, $\rho_k^{(s)} = \frac{||k^{(s)} - k^{(s-1)}||_\infty}{||k^{(s-1)} - k^{(s-2)}||_\infty}$

end while

One can easily show that $\mathbf{L}\phi^{(s)} = q_{source}^{(s)}$ can be solved by the Proper Orthogonal Decomposition Galerkin projection approach. The following pseudo-algorithm shows how this can be applied to the power iterative method 15.

Algorithm 2 Pseudo Algorithm for Power Iterative Method with POD-Galerkin Method

Denote that a reduced order transform operator as $\mathbf{U}$ that is an orthogonal matrix

Set $k_0 = k_{initial}$

Set the initial guess, $\phi^0 = \phi^{initial}$

$s = 0$

Project the matrix operator $\mathbf{L}$ onto the reduced order subspace $\tilde{\mathbf{L}} = \mathbf{U}^T\mathbf{L}\mathbf{U}$

while $||\phi^{(s)} - \phi^{(s-1)}|| > \epsilon_{\phi,1}(1 - \rho_\phi^{(s)})$ or $||k^{(s)} - k^{(s-1)}|| > \epsilon_k(1 - \rho_k^{(s)})$ **do**

$s = s + 1$

Calculate $q_{source}^{(s)} = \frac{1}{k^{(s-1)}}\mathbf{F}\phi^{(s-1)}$

Project the source term onto the reduced order subspace, $\tilde{q}_{source}^{(s)} = \mathbf{U}^T q_{source}^{(s)}$

Solve $\tilde{\mathbf{L}}\tilde{\phi}^{(s)} = \tilde{q}_{source}^{(s)}$.

Project back onto the original space, $\phi^{(s)} = \mathbf{U}^T\tilde{\phi}^{(s)}$

Update k-eigenvalue, $k^{(s)} = k^{(s-1)}\frac{\int dV \nu_f \Sigma_f \phi^{(s)}}{\int_V dV \nu_f \Sigma_f \phi^{(s-1)}}$

Approximate the spectral radius of $\phi^{(s)}$, $\rho_\phi^{(s)} = \frac{||\phi^{(s)} - \phi^{(s-1)}||_\infty}{||\phi^{(s-1)} - \phi^{(s-2)}||_\infty}$

Approximate the spectral radius of $k^{(s)}$, $\rho_k^{(s)} = \frac{||k^{(s)} - k^{(s-1)}||_\infty}{||k^{(s-1)} - k^{(s-2)}||_\infty}$

end while

This approach is very efficient and one can expect to achieve large reduction in computational cost and time. In addition, it is important to note that this method can be applied to the other acceleration techniques commonly applied in Nuclear Engineering. For example, the shifted-inverse power iterative method can be modified with the POD-Galerkin method shown in the following pseudo algorithm.

Algorithm 3 Pseudo Algorithm for Shifted-Inverse Power Iterative Method with POD-Galerkin Method adapted from [7]

Denote that a reduced order transform operator as $\mathbf{U}$

Set a scalar σ that is close to $1/k$

Set $k_0 = k_{initial}$

Set the initial guess, $\phi^0 = \phi^{initial}$

$s = 0$

Project the matrix operator $\mathbf{L}$ onto the reduced order subspace $\tilde{\mathbf{L}} = \mathbf{U}^T\mathbf{L}\mathbf{U}$

while $||\phi^{(s)} - \phi^{(s-1)}|| > \epsilon_{\phi,1}(1 - \rho_\phi^{(s)})$ or $||k^{(s)} - k^{(s-1)}|| > \epsilon_k(1 - \rho_k^{(s)})$ **do**

$s = s + 1$

Calculate $q_{source}^{(s)} = \frac{1}{k^{(s-1)}}\mathbf{F}\phi^{(s-1)}$

Project the source term onto the reduced order subspace, $\tilde{q}_{source}^{(s)} = \mathbf{U}^T q_{source}^{(s)}$

Solve $(\tilde{\mathbf{L}} - \sigma\mathbf{I})\tilde{\phi}^{(s)} = \tilde{q}_{source}^{(s)}$.

Project back onto the original space, $\phi^{(s)} = \mathbf{U}^T\tilde{\phi}^{(s)}$

Update k-eigenvalue, $k^{(s)} = k^{(s-1)}\frac{\int dV \nu_f \Sigma_f \phi^{(s)}}{\int_V dV \nu_f \Sigma_f \phi^{(s-1)}}$

Approximate the spectral radius of $\phi^{(s)}$, $\rho_\phi^{(s)} = \frac{||\phi^{(s)} - \phi^{(s-1)}||_\infty}{||\phi^{(s-1)} - \phi^{(s-2)}||_\infty}$

Approximate the spectral radius of $k^{(s)}$, $\rho_k^{(s)} = \frac{||k^{(s)} - k^{(s-1)}||_\infty}{||k^{(s-1)} - k^{(s-2)}||_\infty}$

end while

The algorithm is almost same with the power iterative method. However, the rate of convergence is dramatically improved if $\sigma \approx 1/k$ and the rate of convergence is given by [7]

$$d = \frac{|1/k_1 - \sigma|}{|1/k_2 - \sigma|} \tag{1.4}$$

where k_1 is the dominant eigenvalue and k_2 is the second dominant eigenvalue. One can also apply the Rayleigh Quotient Iteration since forming the explicit inverse is not expensive in the reduced space. However, these approaches require extra dense matrix-vector multiplications in each iterative step. In addition, modern fast iterative solvers cannot be applied. Therefore, more efficient algorithms for generalized eigenvalue problem are desired.

1.2 Thesis Contents

This thesis develops hybrid reduced order modeling methods to generate surrogates for models described as generalized eigenvalue problems. Chapter 2 briefly describes the neutronics model employed and chapter 3 describes the mathematical framework of ROM methods. Chapter 4 briefly presents theory of the generalized polynomial chaos, sparse approximation and input parameter reduction. Numerical experiments are documented in chapter 5 and 6, and concluding remarks are found in chapter 7.

Chapter 2

Neutronics Problem

In reactor simulation, the determination of the multiplication factor, k, and the distribution of neutrons in space and energy is the fundamental task for reactor engineers because this knowledge helps one sustain and control the fission chain reaction. In reactor physics, two major approaches to determine these quantities are the diffusion approximation and the transport approximations. Neutrons are treated as a diffusive material like gas molecules in the diffusion approximation. In the transport approximations, the distribution of neutrons is described by the Boltzmann equation which is an integro-differential equation, and is much more difficult to solve than diffusion equation. The diffusion approximation can be derived from the transport equation under some simplifying assumptions.

2.1 The Diffusion Approximation

The basics of neutron diffusion theory are well presented in the textbooks from Duderstadt and Hamilton [8] and Stacey [9]. The neutron diffusion equation can be written as:

$$\nabla J(r,E) + \Sigma_t(r,E)\phi(r,E) = \int_0^\infty dE' \Sigma_s(r,E' \to E)\phi(r,E') + \frac{\chi(E)}{k}\int_0^\infty dE' \nu(r,E')\Sigma_f(r,E')\phi(r,E') \quad (2.1)$$

with Fick's law giving:

$$J(r,E) = -D(r,E)\nabla\phi(r,E) \tag{2.2}$$

where r is the position vector, E is the neutron energy, J is the neutron current, $\phi(r,E)$ is the scalar neutron flux, $\Sigma(r,E)$ is the total macroscopic cross-section, $\Sigma_s(r,E' \to E)$ is the macroscopic scattering cross-section, $\chi(E)$ is the fission yield spectrum, $\nu(E')$ is the average number of neutrons born in a fission, $\Sigma_f(r,E)$ is the macroscopic fission cross-section, and $D(r,E)$ is the diffusion coefficients. To solve this equation, one can discretize the equation directly or combine Eq.(2.1) and Eq.(2.2) to obtain the P_0 equation.

In the former case, the neutron energy E is divided into G energy groups and Eq.(2.5) is integrated over each energy group. That can be written as:

$$\nabla J_g + \Sigma_t^g \phi_g = \sum_{g'=1}^{G} \Sigma_s^{g' \to g} \phi_{g'} + \frac{\chi_g}{k} \sum_{g'=1}^{G} \nu_g \Sigma_f^{g'} \phi_{g'}, \quad g = 1, \cdots G \tag{2.3}$$

$$J_g = -D_g \nabla \phi_g, \quad g = 1, \cdots G \tag{2.4}$$

where Eq.(2.3) can be spatially discretized by the finite element or finite difference method with appropriate boundary conditions. For the latter case, Eq.(2.1) and Eq.(2.2) are combined into the P_0 equation:

$$\begin{aligned} -\nabla \cdot D(r,E)\nabla\phi(r,E) + \Sigma_t(r,E)\phi(r,E) = &\int_0^\infty dE' \Sigma_s(r,E' \to E)\phi(r,E') \\ &+ \frac{\chi(E)}{k} \int_0^\infty dE' \nu(r,E')\Sigma_f(r,E')\phi(r,E') \end{aligned} \tag{2.5}$$

then Eq.(2.5) is discretized into the multi-group neutron diffusion equation:

$$-\nabla \cdot D_g \nabla \phi_g + \Sigma_t^g \phi_g = \sum_{g'=1}^{G} \Sigma_s^{g' \to g} \phi_{g'} + \frac{\chi_g}{k} \sum_{g'=1}^{G} \nu_g \Sigma_f^{g'} \phi_{g'}, \quad g = 1, \cdots G \tag{2.6}$$

which can be solved by the finite element method or finite difference method. For the finite difference method, see Stacey [9] and Duderstadt and Hamilton [8] for examples.

The k-eigenvalue diffusion equation can be written in block matrix form. For the P_0 equation, it is given by:

$$\phi = \begin{bmatrix} \phi_1^T & \cdots & \phi_i^T & \cdots & \phi_G^T \end{bmatrix}^T \tag{2.7}$$

where ϕ_i is a vector of discretized scalar flux in energy group. Then, Eq.(2.6) can be rewritten in sparse block matrix form:

$$\mathbf{L}\phi = \lambda \mathbf{F}\phi \tag{2.8}$$

where

$$\mathbf{L} = \begin{bmatrix} \boldsymbol{\nabla} + \boldsymbol{\Sigma}_R^1 & -\boldsymbol{\Sigma}_s^{2\to 1} & \cdots & -\boldsymbol{\Sigma}_s^{G\to 1} \\ -\boldsymbol{\Sigma}_s^{1\to 2} & \boldsymbol{\nabla} + \boldsymbol{\Sigma}_R^2 & \cdots & -\boldsymbol{\Sigma}^{G\to 2} \\ \vdots & \vdots & \ddots & \vdots \\ -\boldsymbol{\Sigma}_s^{1\to G} & -\boldsymbol{\Sigma}_s^{2\to G} & \cdots & \boldsymbol{\nabla} + \boldsymbol{\Sigma}_R^G \end{bmatrix} \tag{2.9}$$

$$\mathbf{F} = \begin{bmatrix} \chi_1\nu_1\boldsymbol{\Sigma}_f^1 & \chi_1\nu_2\boldsymbol{\Sigma}_f^2 & \cdots & \chi_1\nu_G\boldsymbol{\Sigma}_f^G \\ \chi_2\nu_1\boldsymbol{\Sigma}_f^1 & \chi_2\nu_2\boldsymbol{\Sigma}_f^2 & \cdots & \chi_2\nu_G\boldsymbol{\Sigma}_f^G \\ \vdots & \vdots & \ddots & \vdots \\ \chi_G\nu_1\boldsymbol{\Sigma}_f^1 & \chi_G\nu_2\boldsymbol{\Sigma}_f^2 & \cdots & \chi_G\nu_G\boldsymbol{\Sigma}_f^G \end{bmatrix} \tag{2.10}$$

where $\boldsymbol{\nabla}$ is a matrix resulted from the discretization of the ∇ operator on the scalar flux ϕ_i and $\boldsymbol{\Sigma}_R^g$ is a diagonal matrix defined as the group removal cross-section to be $\boldsymbol{\Sigma}_R^g = \boldsymbol{\Sigma}_t^g - \boldsymbol{\Sigma}_s^{g\to g}$ resulting from the discretization. The P_1 equation can also be discretized in a similar manner.

2.2 Newton's Method for k-Eigenvalue Problem

Finding the fundamental eigenpair (eigenvalue and eigenvector) is the computational burden for large scale problem. Historically in nuclear engineering, this generalized eigenvalue problem is solved by the power iterative method, but the rate of convergence is slow for models with the dominance ratio very close to 1.0 [10]. Over the past three decades, a variety of approaches has been proposed to accelerate the convergence of the solution, eg. Wielandt shift approach [11] and Chebyshev acceleration [12]. Recently, the Newton-Krylov approach is beging considered for multi-group neutronics calculations [13]. Newton's method is characterized by its quadratic convergence and difficulty in calculating and inverting a Jacobian matrix to obtain the Newton step.

As shown in the previous section, the multi-group diffusion problem can be written in a generalized eigenvalue form:

$$\mathbf{L}\phi = \lambda \mathbf{F}\phi \tag{2.11}$$

and assume a constraint on the eigenpair denoted by:

$$f(\lambda, \phi) = 0. \tag{2.12}$$

Note that it is common to use $f(\lambda, \phi) = \phi^T\phi - 1$, so this constraint is used in this section. Then, finding eigenpairs is equivalent to minimizing a residual function defined by:

$$R(\lambda, \phi) = \begin{bmatrix} \mathbf{L}\phi - \lambda\mathbf{F}\phi \\ f(\lambda, \phi) \end{bmatrix}. \tag{2.13}$$

When λ and ϕ are eigenpairs, the residual function is equal to zero. This is also considered as an optimization problem, which can be solved by Newton's method. To find a solution, expand Eq.(2.13) around the initial guess λ_0 and ϕ_0:

$$\mathrm{R}(\lambda, \phi) = \mathrm{R}(\lambda_0, \phi_0) + \mathbf{J}(\lambda_0, \phi_0) \begin{bmatrix} \Delta\lambda \\ \Delta\phi \end{bmatrix} + \cdots \tag{2.14}$$

where $\Delta\lambda = \lambda - \lambda_0$, $\Delta\phi = \phi - \phi_0$ and the Jacobian matrix is given by [13]:

$$\mathbf{J}(\lambda, \phi) = \begin{bmatrix} \mathbf{L} - \lambda_0\mathbf{F} & -\mathbf{F}\phi_0 \\ 2\phi_0^T & 0 \end{bmatrix} \tag{2.15}$$

Now, setting $R(\lambda, \phi) = 0$, the equation for the Newton step is given by:

$$-\mathrm{R}(\lambda_0, \phi_0) = \mathbf{J}(\Delta\lambda, \Delta\phi) \begin{bmatrix} \Delta\phi_0 \\ \Delta\lambda_0 \end{bmatrix} \tag{2.16}$$

Therefore, each Newton step is given by:

$$\begin{bmatrix} \Delta\phi_{i+1} \\ \Delta\lambda_{i+1} \end{bmatrix} = \mathbf{J}^{-1}(\lambda_i, \phi_i)\mathrm{R}(\lambda_i, \phi_i) \tag{2.17}$$

It is important to note that the exact Newton's method is computationally prohibitive to calculate. Therefore, the inverse of the Jacobian matrix is calculated by iterative methods such as the Krylov subspace method or a fixed-point iteration. The efficiency of Krylov subspace method is governed by a preconditioner and incomplete LU factorization is typically utilized. In addition, a reordering technique must be applied to obtain precon-

ditioners efficiently. Otherwise, the incomplete factorization takes long time and number of non-zero elements in preconditioners is extremely large to store. The study on the Newton's method in Nuclear Engineering application is a quite recent topic (See [14] for the details).

2.3 Transport Calculation by T-NEWT sequence

The neutron transport equation is more accurate compared with the diffusion approximation. The general form of the neutron transport equation is given as:

$$\hat{\Omega} \cdot \vec{\nabla}\psi(r, \hat{\Omega}, E) + \Sigma_t(\vec{r}, E)\psi(\vec{r}, \hat{\Omega}, E) = Q(\vec{r}, \hat{\Omega}, E) \tag{2.18}$$

where $\psi(\vec{r}, \hat{\Omega}, E)$ angular flux, Σ_t total cross-section, and $Q(\vec{r}, \hat{\Omega}, E)$ source term. The source term Q is composed of three components:
1. a scattering source:

$$S(\vec{r}, \hat{\Omega}, E) = \int_{4\pi} d\hat{\Omega}' \int_{4\pi} dE' \Sigma_s(\vec{r}, \hat{\Omega}' \to \hat{\Omega}, E' \to E)\psi(\vec{r}, \hat{\Omega}', E') \tag{2.19}$$

2. a fission source:

$$F(\vec{r}, \hat{\Omega}, E) = \frac{\chi(\vec{r}, E)}{4\pi} \int_0^{\infty} dE' \nu_f(\vec{r}, E')\Sigma_f(\vec{r}, E')\phi(\vec{r}, E') \tag{2.20}$$

and 3. an external source or fixed source: $S_{\text{ext}}(\vec{r}, E)$. To formulate the k-eigenvalue problem, recast Eq.(2.18) into an equation of the form:

$$\begin{aligned} \hat{\Omega} \cdot \vec{\nabla}\psi(r, \hat{\Omega}, E) + \Sigma_t(\vec{r}, E)\psi(\vec{r}, \hat{\Omega}, E) = & \\ \int_{4\pi} d\hat{\Omega}' \int_{4\pi} dE' \Sigma_s(\vec{r}, \hat{\Omega}' \to \hat{\Omega}, E' \to E)\psi(\vec{r}, \hat{\Omega}', E') & \\ + \frac{\chi(\vec{r}, E)}{4\pi k} \int_0^{\infty} dE' \nu_f(\vec{r}, E')\Sigma_f(\vec{r}, E')\phi(\vec{r}, E') & \end{aligned} \tag{2.21}$$

T-NEWT is TRITON sequence to solve 2D transport equation by NEWT in SCALE developed by Oak Ridge National Laboratory. NEWT is a 2D multi-group discrete-ordinates radiation transport code with complex geometry developed by Oak Ridge National Laboratory. T-NEWT is capable of solving for the k-eigenvalue. Sample input files

for T-NEWT sequence can be found in the appendix.

2.4 Model Description

For uncertainty quantification, five different neutronics problems are considered. Each model is assumed to have uncertainties in cross-section data or atomic density values. Each model has a different number of stochastic variables.

A. 1D Simple Fuel Model

Case A is a simple two-group neutron diffusion equation that consists of uranium (left) and water (right) with equal widths of $w = 50$ cm. The two-group diffusion equation for this model is given by

$$\begin{aligned} &- D_1(x,\xi)\frac{\partial^2}{\partial x^2}\phi_1(x,\xi) + \Sigma_R^1(x,\xi)\phi_1(x,\xi) \\ &\qquad = \frac{1}{k}\chi_1(x,\xi)\nu_1(x,\xi)\Sigma_f^1(x,\xi)\phi_1(x,\xi) + \frac{1}{k}\chi_1(x,\xi)\nu_2(x,\xi)\Sigma_f^2(x,\xi)\phi_2(x,\xi) \end{aligned}$$

$$-D_2(x,\xi)\frac{\partial^2}{\partial x^2}\phi_2(x,\xi) + \Sigma_R^2(x,\xi)\phi_2(x,\xi) - \Sigma_s^{1\to 2}(x,\xi)\phi_1(x,\xi) = 0$$

where the induced neutrons are assumed to be born only in the energy group 1. Fig. 2.1 shows the geometry of this simple fuel model. A Neumann boundary condition is assigned to the left boundary whereas a Dirichlet boundary condition is assigned to the right boundary.

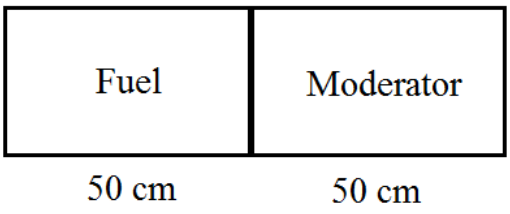

Figure 2.1: The Geometry of the Simple Fuel Model

The boundary conditions are described as:

$$\begin{aligned} \left.\frac{d}{dx}\phi_g\right|_{x=0} &= 0 \\ \phi_g|_{x=2w} &= 0 \end{aligned}$$

In addition, a constraint is imposed on the fluxes:

$$\sum_{g=1}^{2} \phi_g^T \phi_g = 1$$

Then, this model is discretized into generalized eigenvalue problem of the form:

$$\mathbf{L}\phi = \frac{1}{k}\mathbf{F}\phi$$

where $\phi = \begin{bmatrix} \phi_1^T & \phi_2^T \end{bmatrix}^T$ where ϕ_i is a vector of discretized flux of the energy group i. Table 2.1 shows the cross-section data for this model. To simulate uncertainty in this model, the cross-section data are assumed to have uncertainty in the values.

Table 2.1: Test Model A Cross-Section Data [8]

	Fuel		Moderator	
	1 of 2	2 of 2	1 of 2	2 of 2
$\nu\Sigma_f$	.008476	.18514	0.0	0.0
Σ_a	0.01207	0.1210	0.0004	0.0197
D	1.2627	.3543	1.13	0.16
Σ_R	0.02619	0.1210	0.0494	0.0197

The thermal group absorption cross-section and the removal cross-section are assumed to be equal, and hence the total number of stochastic variables is 12. The model is spatially discretized with equal node width resulting in 1000 total nodes, and hence the dimension of L and F is 2000 (because the number of unknowns per node is equal to two, representing the two-group flux).

Fig. 2.2 shows the fast group neutron flux distribution and Fig. 2.3 shows the thermal

group neutron flux distribution with the reference cross-section values.

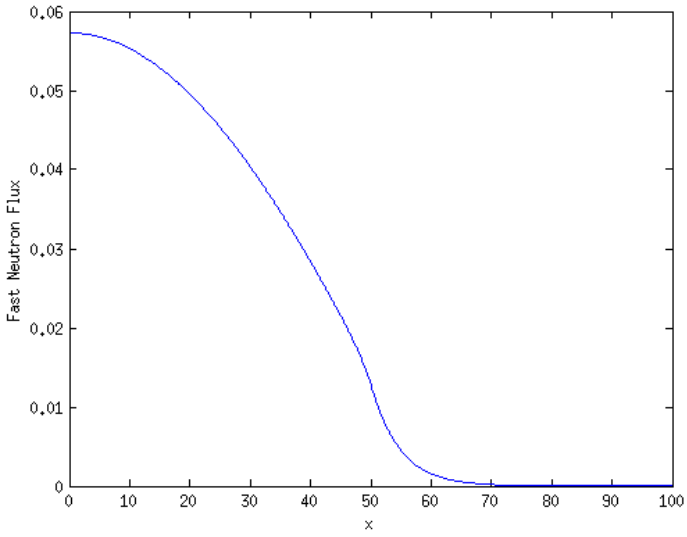

Figure 2.2: Fast Group Flux Distribution

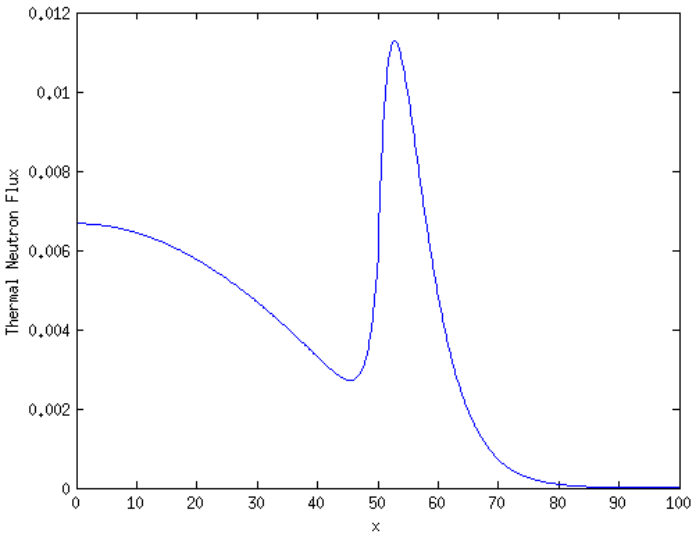

Figure 2.3: Thermal Group Flux Distribution

B. 1D Assembly Model

The test model B is a two-group neutron diffusion model describing a 1-D fuel assembly model consisting of MOX and UO_2 with reflective boundary conditions at both sides [15]. The width of each assembly is 10 cm and consists of 8 fuel pins surrounded on both sides by water that are spatially discretized with equal node width. Fig. 2.4 and Fig. 2.5 show the geometry of this model.

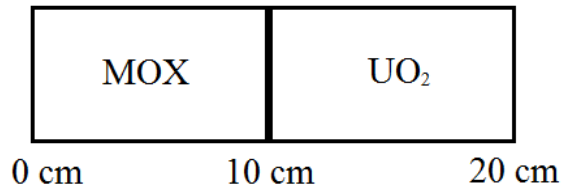

Figure 2.4: The Geometry of Assembly

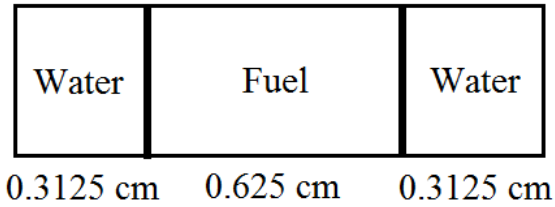

Figure 2.5: The Geometry of Pin Cell

The form of the neutron diffusion equation for this model is given by:

$$\frac{d}{dx}J^g(x,\xi) + \Sigma_R^g(x,\xi)\phi_g(x,\xi) =$$
$$\sum_{g'=1,g'\neq g}^{2} \Sigma_s^{g's\to g}(x,\xi)\phi_{g'}(x,\xi) + \frac{\chi_g(x,\xi)}{k}\sum_{g'=1}^{2}\nu_f^{g'}(x,\xi)\Sigma_f^{g'}(x,\xi)\phi_{g'}(x,\xi)$$

with Fick's law giving an expression for the neutron current:

$$J^g(x,\xi) = -D_g(x,\xi)\frac{d}{dx}\phi_g(x,\xi)$$

Then, discretize the assembly into I equal width cells where each cell is defined for $x \in [x_i, x_{i+1}]$ and the value of the cell edge locations is given by:

$$x_i = \frac{L}{I}(i-1)$$

Then, this diffusion equation can be discretized using the cell-edge current $J_{g,i}$, the cell-edge flux $\phi_{g,i}$, and the cell-average flux $\bar{\phi}_{g,i}$:

$$\begin{aligned}
J_{g,i+1} - J_{g,i} + \Sigma_R^g \bar{\phi}_{g,i} h - \sum_{g'=1, g' \neq q}^{2} \Sigma_s^{g' \to g} \bar{\phi}_{g,i} h &= \frac{\chi_g}{k} \sum_{g'=1}^{2} \nu_f^{g'} \Sigma_f^{g'} \bar{\phi}_{g,i} h \\
D_{g,i}(\phi_{g,i+1} - \bar{\phi}_{g,i}) + \frac{1}{2} J_{g,i+1} h &= 0 \\
D_{g,i}(\phi_{g,i} - \bar{\phi}_{g,i}) + \frac{1}{2} J_{g,i} h &= 0 \\
\bar{\phi}_{g,1} - \phi_{g,1} &= 0 \\
\phi_{g,I+1} - \bar{\phi}_{g,I} &= 0
\end{aligned}$$

where $h = x_{i+1} - x_i$ and the cell-average flux is defined as:

$$\bar{\phi}_{g,i} = \int_{x_i}^{x_{i+1}} \phi_g(x) dx$$

Then, the model is simplified into generalized eigenvalue problem of the form:

$$\mathbf{L}x = \frac{1}{k}\mathbf{F}x$$

where $x = \begin{bmatrix} J_1^T & J_2^T & \phi_1^T & \phi_2^T & \bar{\phi}_1^T & \bar{\phi}_2^T \end{bmatrix}^T$ which represent the discretized quantities. Table 2.2 shows the cross-section data for this model. The cross-section data are assumed to have uncertainty in their values.

Table 2.2: Test Model B Cross-Section Data [15]. Σ_f^1, ν_f^1, $\bar{\mu}_0^1$, $\Sigma_s^{2\to1}$, χ^1, $\bar{\mu}_0^2$ are all zero.

Cross-Sections	Σ_t^1	$\Sigma_s^{1\to1}$	$\Sigma_s^{1\to2}$	χ_1	Σ_t^2	$\Sigma_s^{2\to1}$	Σ_f^2	ν_f^2
MOX fuel	0.2	0.185	0.015	1	1.2	0.9	0.3	1.5
Uranium fuel	0.2	0.185	0.015	1	1.0	0.9	0.1	1.5
Water	0.2	0.17	0.03	0	1.1	1.1	0	0

with a normalization constraint

$$h\sum_{g=1}^{2}\bar{\phi}_g^T\bar{\phi}_g = 1$$

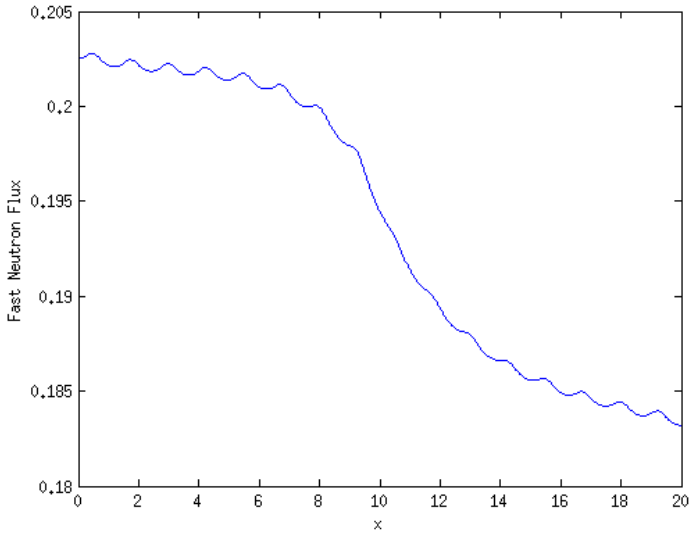

Figure 2.6: Fast Group Flux Distribution

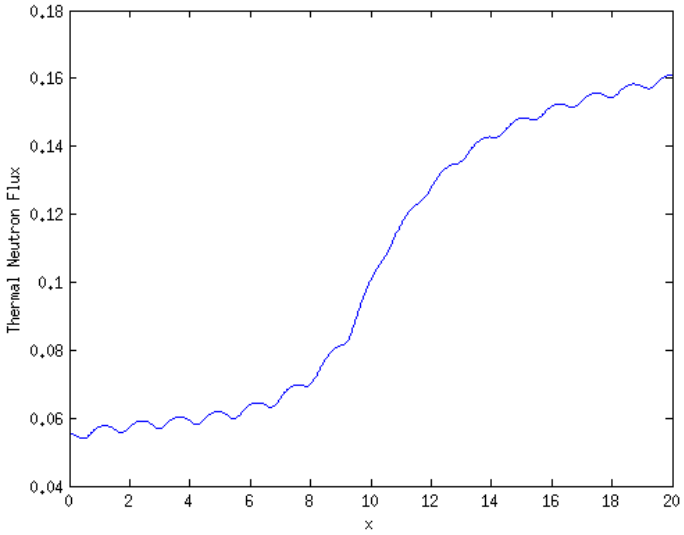

Figure 2.7: Thermal Group Flux Distribution

Fig. 2.6 and Fig. 2.7 show the fast and thermal neutron flux distribution for the reference cross-section values.

C. 2D MOX Fuel Calculation

The test model C is a 2D transport model describing a fresh MOX fuel from the Phase IV-A Burn-up Credit Benchmark by OECD/NEA Burn-up Credit Working Group [16]. The geometry for this model is an infinite PWR fuel cell lattice as shown in Fig. 2.8. The Fuel pin pitch is 1.33 cm square pin array, the fuel pellet radius is 0.412 cm and the cladding thickness is 0.0063 cm. In this model, it is assumed that no air or helium gas is present.

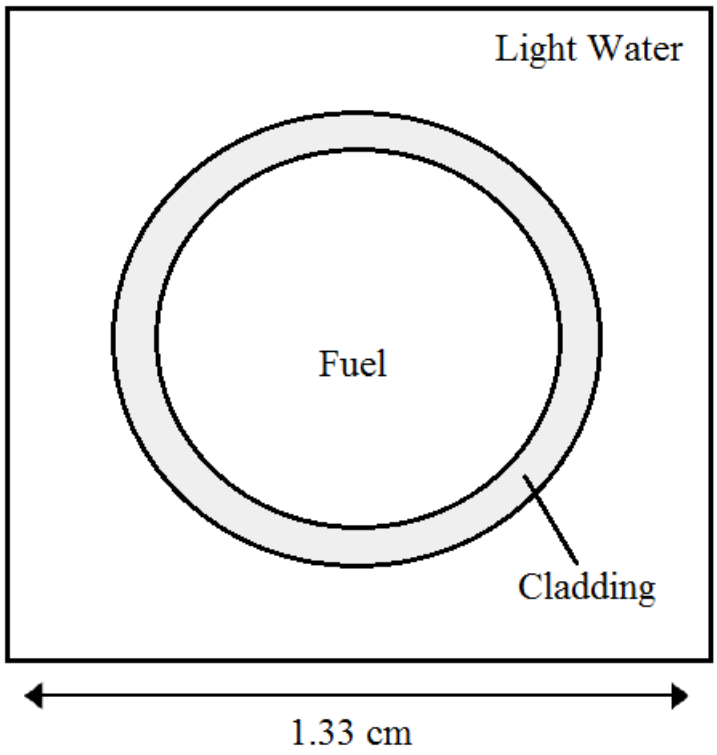

Figure 2.8: Geometry of MOX fuel pin cell for PWR

In reference [16], three different types of MOX fuels are considered. However, only the case A, a typical plutonium composition for material derived from the reprocessing of thermal reactor UO_2 fuel, is considered. Table 2.3-Table 2.4 show the material compositions for Case A. These models are evaluated using the SCALE code package with END/F V 44 energy group cross-section data with CENTRAM. Appendix A shows the input file for the T-NEWT sequence. In the original benchmark problem, the discrete ordinates calculations were performed using S_8 quadrature and P_3 scattering for all mixture. However, in uncertainty quantification, many model evaluations are necessary and the calculation is reduced by simplying scattering order to 2 in all moderator materials and 1 for fuels and structural materials.

Table 2.3: Non-fission material compositions [16]

Nuclide	Atoms/Barn.cm
Zr	4.2982E-2
Fe	1.4838E-4
Cr	7.5891E-5
H	6.6724E-2
O	3.3362E-2

Table 2.4: Material Compositions for Fresh MOX Case A [16]

Nuclide	Atoms/Barn.cm	Nuclide	Atoms/Barn.cm
^{234}U	2.7999E-7	^{235}U	1.4838E-4
^{238}U	2.3074E-2	^{238}Pu	2.4700E-5
^{239}Pu	8.0623E-4	^{240}Pu	3.1298E-4
^{241}Pu	1.6533E-5	^{242}Pu	5.3981E-5
^{16}O	4.8992E-2		

D. 2D Quarter Assembly Calculation

The test model D is 2D 8×8 quarter assembly model. Appendix A shows the sample input file for this model. This problem can be found in the SCALE sample problem directory. However, the fuel composition is modified into MOX fuel as described in the test case C. Fig. 2.9 shows the geometry of the quarter assembly models.

Fig. 2.10-2.11 show the selected output from the test problem D. Detailed material composition and geometric information can be found in the sample input in Appendix A. Fig. 2.10 shows the 1st energy group scalar flux and Fig. 2.11 shows the 44th energy group scalar flux.

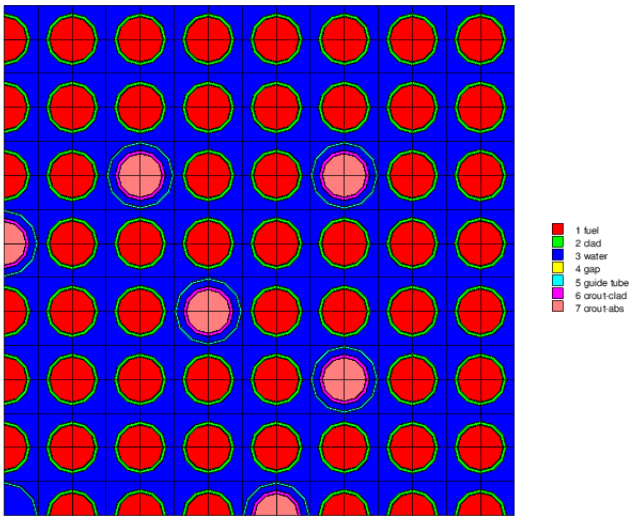

Figure 2.9: Geometry of Quarter Assembly Model

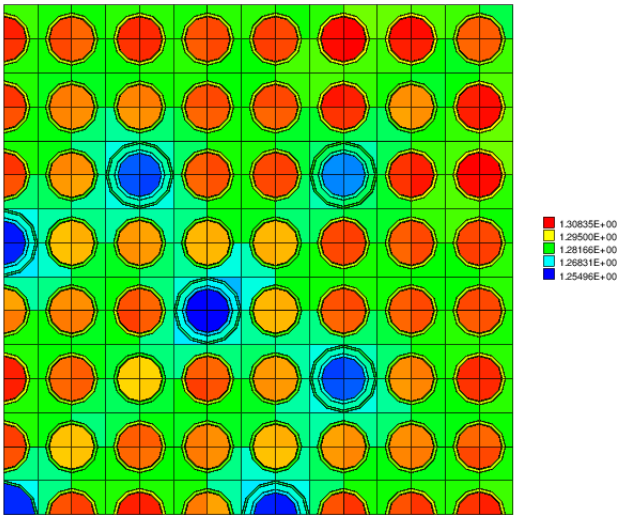

Figure 2.10: Group 1 Flux

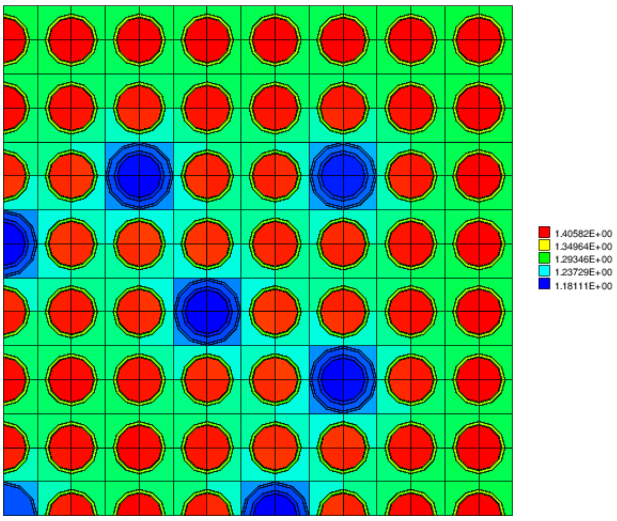

Figure 2.11: Group 44 Flux

E. 2D Pin-Cell Model by Diffusion Approximation

The last model to be considered is a pin-cell model using the 44 group finite element diffusion equation. The geometry for this model is an infinite PWR fuel cell lattice as shown in Fig. 2.12. The Fuel pin pitch is 1.3127 cm square pin array, the fuel pellet radius is 0.410 cm and the cladding thickness is 0.0065 cm. No air/helium gap is assumed in these models. Different from model C, the MOX fuel is irradiated with 40GWd/teHM followed by one year of cooling in water. Table 2.5 shows the reference values of the materials of this model. Each atomic density is assumed to have uncertainty in its value within $\pm$ 10% of it reference value. In addition, microscopic cross-section values are also assumed to have uncertainty within $\pm$ 5% of their reference values.

Table 2.5: Material Compositions for MOX Case A (40GWd/teHM) with one year cooling

Nuclide	Atoms/Barn.cm	Nuclide	Atoms/Barn.cm
^{234}U	7.7718E-7	^{235}U	2.9018E-5
^{236}U	6.1753E-6	^{238}U	2.2365E-2
^{238}Pu	2.5504E-5	^{239}Pu	4.5028E-4
^{240}Pu	2.9067E-4	^{241}Pu	1.8125E-4
^{242}Pu	9.1733E-5	^{237}Np	3.0746E-6
^{241}Am	2.4023E-5	^{243}Am	2.4023E-5
^{16}O	4.8992E-2	^{242}Cm	6.7186E-7
^{243}Cm	6.7186E-7	^{244}Cm	1.3749E-5
^{245}Cm	1.6967E-6	^{95}Mo	4.4441E-5
^{99}Tc	5.3736E-5	^{101}Ru	5.774E-5
^{103}Rh	4.9708E-5	^{109}Ag	1.1408E-5
^{133}Cs	5.7100E-5	^{143}Nd	3.8610E-5
^{145}Nd	2.8038E-5	^{147}Sm	5.2265E-6
^{149}Sm	3.3504E-7	^{150}Sm	1.3820E-5
^{151}Sm	1.5043E-6	^{152}Sm	6.2143E-6
^{153}Eu	7.5074E-6	^{155}Gd	1.4403E-7

Fig. 2.12 shows the triDelaunay mesh used for this model. The number of elements in this triDelaunay mesh is 1703. Therefore, the dimension of $\mathbf{L}$ and $\mathbf{F}$ is 74932. The number of non-zero elements is $\mathbf{L}$ is 12869931 and that of $\mathbf{F}$ is 7120608[1]. Fig. 2.13-2.14 show sparsity patterns of $\mathbf{L}$ and $\mathbf{F}$. Fig. 2.15-2.16 show the sparsity of the reordered $\mathbf{L}$ and $\mathbf{F}$. It is obvious that $\mathbf{L}$ and $\mathbf{F}$ are highly sparse and reorder $\mathbf{L}$ and $\mathbf{F}$ saves large amount of time in incomplete factorization. Fig. 2.17-2.18 show the selected fluxes calculated by the reference value. Using MATLAB, this large problem can be solved within 40 seconds using the reverse symmetric Cuthill-Mckee reordering with GMRES.

[1] 1.4GB in total.

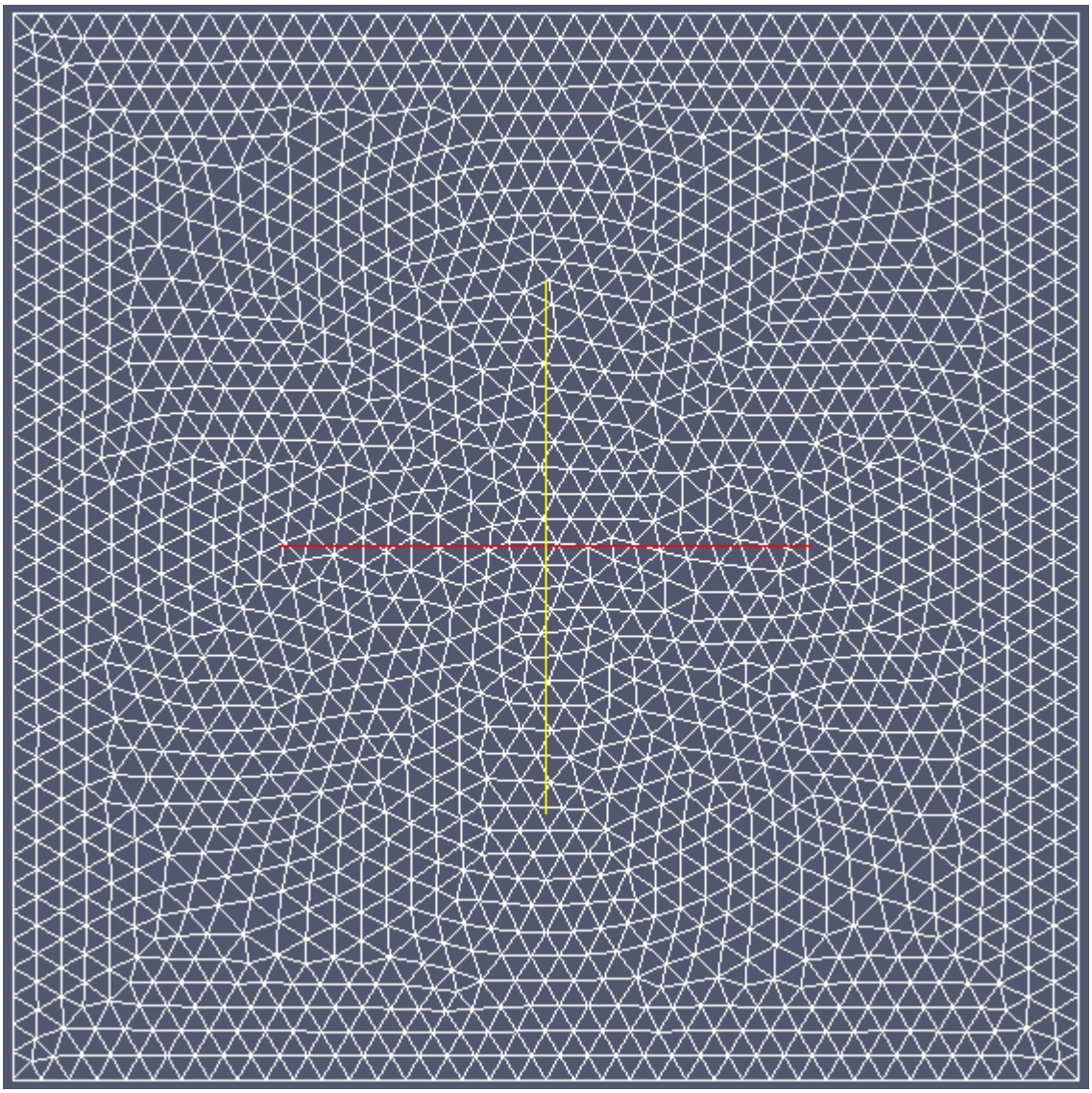

Figure 2.12: triDelaunay Mesh for Pin-cell Model

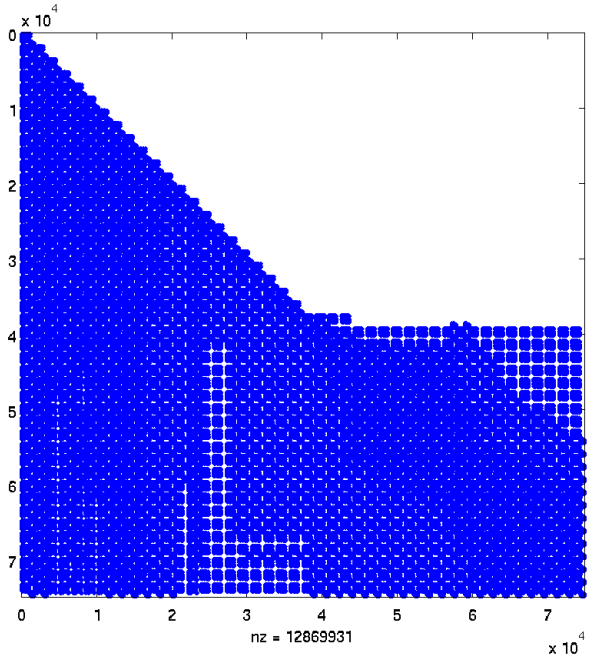

Figure 2.13: Sparsity of **L**

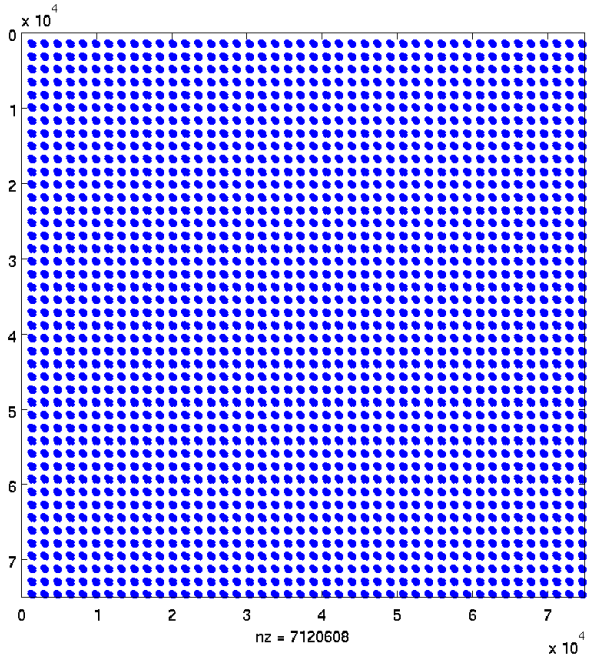

Figure 2.14: Sparsity of **F**

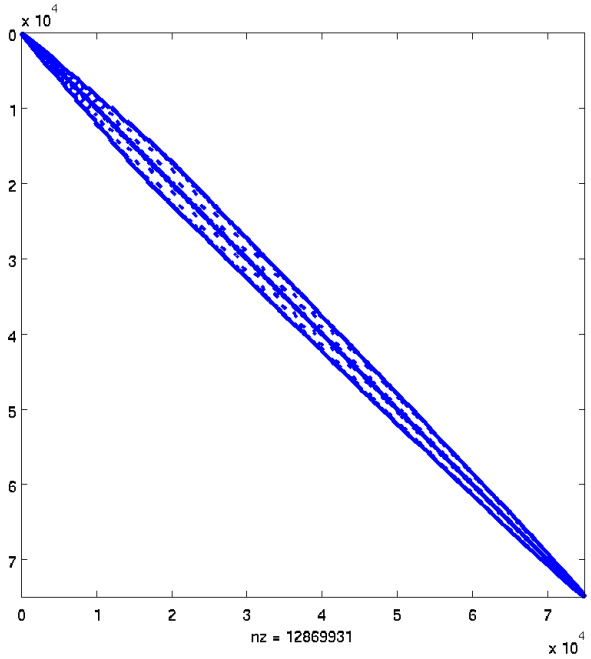

Figure 2.15: Sparsity of reordered $\mathbf{L}$

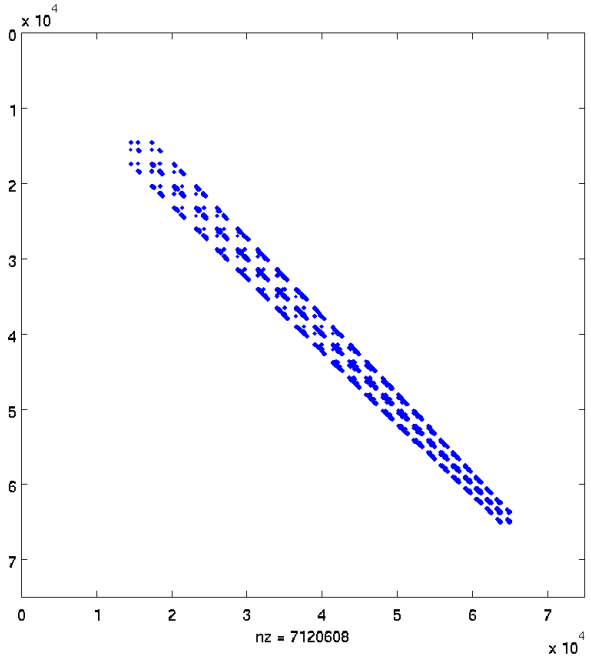

Figure 2.16: Sparsity of reordered $\mathbf{L}$

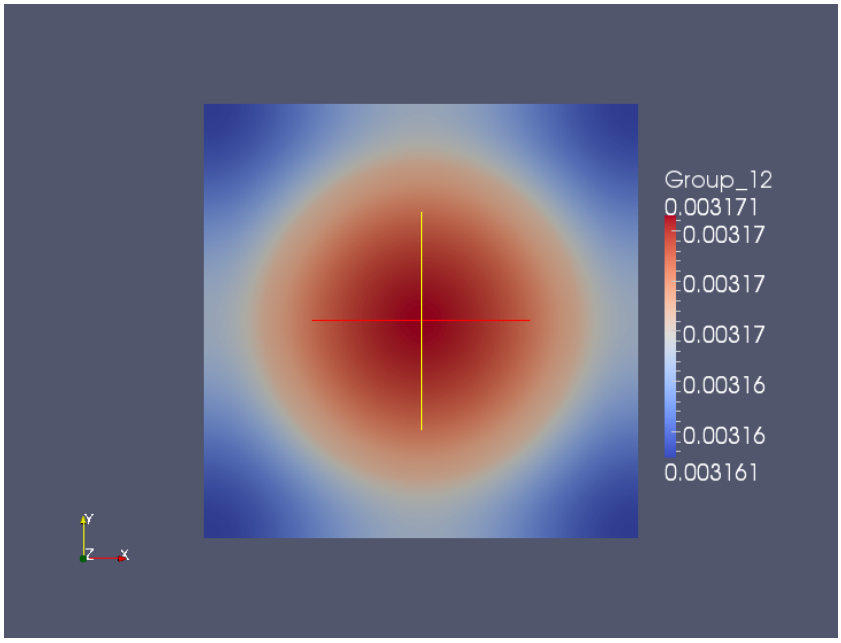

Figure 2.17: 12th Energy Group Flux

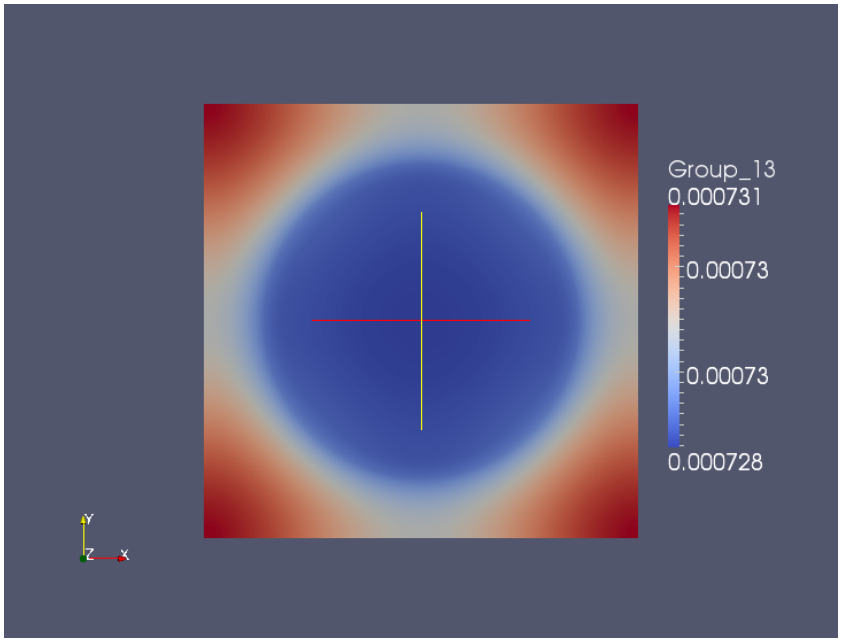

Figure 2.18: 13th Energy Group Flux

Chapter 3

Subspace Method for Reduced Order Modeling

The computational cost of the simulation of the nuclear reactor system has significantly increased over the past few decades as accuracy requirements have become much more stringent to gain both regulatory and public acceptance. There are currently several initiatives around the country aiming to develop advanced codes for reactor simulation. These codes are expected to be high dimensional and expensive to run. This implies that surrogate model techniques must be improved to allow one execute engineering applications such as uncertainty quantification in a practical manner.

In regard to the neutronics problem, the k-eigenvalue problem is the most expensive part of the analysis. Classically, finding the fundamental eigenpair (eigenvector and eigenvalue) is done by the power iterative method which converges slowly for the models associated with the dominance ratio very close to 1.0 [10]. Over the past three decades, a variety of approaches have been proposed to accelerate the convergence of the solution such as the Wielandt shift approach [11] and Chebyshev acceleration [12]. However, as the engineering model becomes more detailed, the computational requirements increase at quadratic rate for memory and cubic rate for computations for the dense matrix. This makes the execution of the high fidelity models computationally prohibitive unless one has access to High Performance Computing facilities.

To overcome the challenge, reduced order modeling (ROM) techniques have been promoted over the past few decades. It is important to note that ROM is referred to differently in different disciplines. For example, Proper Orthogonal Decomposition, Prin-

cipal Component Analysis, Snapshot Method, model order reduction and row-rank approximation are all examples of ROM techniques. The objective of the ROM techniques is to reduce the computational burden without losing the accuracy of original models with a known error bound if possible. In ROM, the dimension of the original model is reduced to a low-dimensional problem that can be executed with lower computational cost. With this reduction, the reduced problem can be executed many times for uncertainty quantification and sensitivity analysis.

Two basic approaches have been applied for surrogate model construction. One of them is a spectral approach and the other is the input parameter reduction. In the former approach, the response is expressed as a linear combination of arbitrary orthogonal basis functions, e.g., trigonometric functions and Legendre polynomials. The generalized Polynomial Chaos (gPC) expansion is considered to be a spectral method. Note that many functions other than orthogonal basis functions can also be used. For example, the sparse grid approximation expands the response as a linear combination of hierarchy hat basis functions.[17]. The spectral approach has the desirable property that the error decays at the exponential rate [2].

In the second approach, the effective number of input parameters is reduced by the finding an active subspace in the input parameter space. This is possible because for many realistic models the response depends on a few degrees of freedom (DOF), where each degree of freedom represents a function of all the parameters combined. If linear functions are employed to describe the relationship between the active DOFs and original input parameters, the ROM problem reduces to a linear algebra problem, where the active DOFs are described by a subspace. The implication is that parameter variations orthogonal to this subspace will produce negligible variations in the response subspace [18, 19].

In this chapter, recently developed and well-established approaches for ROM are briefly reviewed. In addition, ROM techniques for the generalized eigenvalue problem are introduced. The basic idea is to assume that the solution of the forward and the adjoint problem can be expressed as linear combination of r snapshots of the stochastic partial differential equation (SPDE) [20].

3.1 Method of Snapshot

In chapter 2, the multi-group neutron diffusion equation is expressed as a generalized eigenvalue problem of the form $\mathbf{L}\phi = \lambda\mathbf{F}\phi$. Now, assume that the multi-group neutron diffusion equation has uncertainties in cross-section values and atomic densities. Then, represent the uncertainties by a stochastic variable vector ξ where ξ is $s \times 1$ and s is the number of uncertain parameters. Now, insert ξ into a generalized eigenvalue problem,

$$\mathbf{L}(\xi)\phi(\xi) = \lambda(\xi)\mathbf{F}(\xi)\phi(\xi) \tag{3.1}$$

which is the stochastic generalized eigenvalue problem (SGEP) discretized from SPDE and further assume that the dimension of $\phi(\xi)$ is $n \times 1$. For the high fidelity model, solving the SGEP multiple times is computationally expensive and a surrogate model with high accuracy is required. This can be achieved by employing Proper Orthogonal Decomposition (POD) to obtain an optimal low dimensional basis to transform the high dimensional problem into the low dimensional one: see [4, 5] for a more detailed introduction. In other words, the goal is to represent the solution to SGEP using a POD-basis:

$$\phi(\xi) \approx \sum_{i=1}^{r} \alpha_i(\xi)\Phi_i \tag{3.2}$$

where Φ_i are the POD basis, $\alpha_i(\xi)$ are undetermined coefficients, and r is the dimension of the POD basisdesired to be as small as possible. In addition, another important objective of the POD-based method is to find the error bound ϵ of the form:

$$\left|\left|\phi(\xi) - \sum_{i=1}^{r} \alpha_i(\xi)\Phi_i\right|\right| \leq \epsilon \tag{3.3}$$

where ϵ is an upper bound of the errorwhere the undetermined coefficients, $\alpha(\xi)$, are determined via a minimization problem. Unfortunately, the error bound for this form is not known for the discrete case. Instead, it is common to use the following expansion:

$$\sum_{j=1}^{m} \left|\left|\phi(\xi^j) - \sum_{i=1}^{r} \alpha_i(\xi^j)\Phi_i\right|\right|_X^2 \leq \epsilon, \tag{3.4}$$

where X denotes the Hilbert space of the spatial variables and ξ^j denotes the j-th randomly sampled stochastic variables. To determine Φ, multiple approaches are proposed: the method of snapshot with 1. eigenvalue decomposition [21], 2. singular-value decomposition (SVD) [22], and 3. the range-finding algorithm [23].

The method of snapshots [24] is the POD procedure to express a solution to the SGEP by a linear combination of snapshots $\phi(\xi_i)$ denoted by:

$$\phi(\xi) \approx \sum_{j=1}^{m} \beta_j(\xi)\phi(\xi^j) \tag{3.5}$$

where $\beta_i(\xi)$ are undetermined coefficients. In other words, a solution to SGEP is spanned by snapshots described as:

$$\phi(\xi) \in \text{span}(\phi(\xi^1), \cdots, \phi(\xi^i), \cdots, \phi(\xi^m)). \tag{3.6}$$

If $\Phi(\xi^j) = \phi(\xi^j)$ in Eq.(3.4) and $r = m$, then Eq.(3.4) has $\epsilon = 0$. However, the primary objective is to find Φ_i with $r << n$ while minimizing ϵ.

3.1.1 Eigenvalue Decomposition

Eigenvalue decomposition is one possible method to accomplish this goal. Assume a matrix $\mathbf{Y}$ whose columns are known snapshots of the form:

$$\mathbf{Y} = \begin{bmatrix} \phi(\xi^1) & \cdots & \phi(\xi^m) \end{bmatrix} \tag{3.7}$$

where the dimension $\mathbf{Y}$ is $n \times m$. Now, the expression in this form becomes:

$$\mathbf{Y}^T\mathbf{Y}x_i = \mu_i x_i \tag{3.8}$$

where x_i and μ_i are the i-th eigenpair for the symmetric matrix $\mathbf{Y}^T\mathbf{Y}$. Since $\mathbf{Y}^T\mathbf{Y}$ is a real-symmetric matrix, μ_i are always non-negative. Without loss of generality, the eigenvalues μ_i are assumed to be descending order, $\mu_1 \geq \cdots \geq \mu_m$. The method of snapshots shows that an optimal error can be obtained by the POD-basis expressed as [21]:

$$\Phi_i = \frac{1}{\sqrt{\mu_i}}\mathbf{Y}x_i \tag{3.9}$$

and the optimal error bound for Eq.(3.4) is given by [21]:

$$\frac{1}{m}\sum_{j=1}^{m}\left|\left|\phi(\xi^j)-\sum_{i=1}^{r}\langle\Phi_i,\phi(\xi^j)\rangle\Phi_i\right|\right|_X^2=\sum_{i=r+1}^{m}\mu_i \tag{3.10}$$

where $\langle,\rangle$ is the inner product defined as:

$$\langle a,b\rangle = a^T b. \tag{3.11}$$

3.1.2 Singular Value Decomposition

The same error bound can be obtained by the singular value decomposition (SVD). The error bound for SVD is derived using the error bound by the eigenvalue decomposition using the relation between the singular values and eigenvalues. More formal derivation for the error bound of Eq.(3.4) with the singular value decomposition can be found in [22].

SVD is one of the most important tools in numerical linear algebra due to its broad applications in theory and engineering. The SVD is defined as the following manner:

Let $\mathbf{A}\in\mathbb{C}^{m\times n}$. Then reduced SVD of $\mathbf{A}$ is given by:

$$\mathbf{A}=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{3.12}$$

where

$$\mathbf{\Sigma}=\begin{bmatrix}\sigma_1 & & \\ & \ddots & \\ & & \sigma_n\end{bmatrix} \tag{3.13}$$

with $\sigma_1\geq\cdots\geq\sigma_{\min(m,n)}\geq 0$ and with $\mathbf{U}\in\mathbb{C}^{m\times m}$ and $\mathbf{V}\in\mathbb{C}^{n\times n}$ are unitary. Now, the error bound for Eq.(3.4) can be obtained using this definition.

Assume an eigenvalue decomposition for $\mathbf{Y}^*\mathbf{Y}$ implying $\mathbf{Y}^*\mathbf{Y}x_i=\mu_i x_i$ and SVD for $\mathbf{Y}$, $\mathbf{Y}=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$. Then, one can obtain the relation between singular values and eigenvalues:

$$\mu_i=\sigma_i^2. \tag{3.14}$$

This can be shown as the following. Since $\mathbf{Y}^*\mathbf{Y}$ is a real symmetric matrix, we have an

eigenvalue decomposition of the form:

$$\mathbf{Y}^*\mathbf{Y} = \mathbf{X}\mathbf{M}\mathbf{X}^* \tag{3.15}$$

where $\mathbf{X}$ is an orthogonal matrix and $\mathbf{M}$ is a real diagonal matrix with the eigenvalues of $\mathbf{Y}$ on the diagonal. Likewise, using SVD, $\mathbf{Y}^*\mathbf{Y}$ can be factored as:

$$\mathbf{Y}^*\mathbf{Y} = (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^*)^*\mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^* \tag{3.16}$$

Since $\mathbf{V}$ is unitary and $\mathbf{\Sigma}^2$ is a real diagonal matrix, implying that $\mathbf{\Sigma}^2 = \mathbf{M}$. Hence,

$$\mu_i = \sigma_i^2$$

and $\mathbf{V} = \mathbf{X}$. Now, Eq.(3.10) can be written using SVD. Since $\mathbf{V} = \mathbf{X}$, it is obvious that $x_i = v_i$. Now, Eq.(3.9) can be written by the singular value and left singular vector:

$$\Phi_i = \frac{1}{\sqrt{\mu_i}}\mathbf{Y}x_i = \frac{1}{\sigma_i}\mathbf{Y}v_i = \frac{1}{\sigma_i}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^*v_i = u_i \tag{3.17}$$

Then, Eq.(3.10) is written as:

$$\frac{1}{m}\left|\left|\phi(\xi^j) - \sum_{i=1}^{r}\langle\Phi_i, \phi(\xi^j)\rangle\Phi_i\right|\right|_X^2 = \sum_{j=r+1}^{m}\mu_i = \sum_{j=r+1}^{m}\sigma_j^2 \tag{3.18}$$

Therefore, Eq.(3.18) gives an equivalent error bound for SVD. Regardless of whether we choose SVD or eigenvalue decomposition, both forms have the same error bound.

However, the error bound given by Eq.(3.4) is inconvenient in some cases. Therefore, we propose a slightly different error bound:

$$\left|\left|\phi(\xi^j) - \sum_{i=1}^{r}\langle\Phi_i, \phi(\xi^j)\rangle\Phi_i\right|\right|_2 \leq \epsilon \quad \text{for } \forall j \tag{3.19}$$

The optimal solution to Eq.(3.19) is not known, but the upper bound can be obtained with SVD given by:

$$\left|\left|\phi(\xi^j) - \sum_{i=1}^{r}\langle\Phi_i, \phi(\xi^j)\rangle\Phi_i\right|\right|_2 \leq \sigma_{r+1} \quad \text{for } \forall j. \tag{3.20}$$

Proof. Assume an SVD of $\mathbf{Y}$, $\mathbf{Y} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$ where $\mathbf{U} \in \mathbb{R}^{m\times m}$, $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$, and $\mathbf{V} \in \mathbb{R}^{n\times n}$. Then, Eq.(3.19) is written as:

$$\left|\left|\phi(\xi^j) - \sum_{i=1}^{r}\langle\Phi_i, \phi(\xi^j)\rangle\Phi_i\right|\right|_2 = \left|\left|\phi(\xi^j) - \mathbf{U}_r\mathbf{U}_r^*\phi(\xi^j)\right|\right|_2 \tag{3.21}$$

where $\mathbf{U}_r$ represents the first r columns of $\mathbf{U}$ denoted as:

$$\mathbf{U}_r = \begin{bmatrix} u_1 & u_2 & \cdots & u_r \end{bmatrix} \tag{3.22}$$

and $\mathbf{U}_{n-r}$ as:

$$\mathbf{U}_{n-r} = \begin{bmatrix} u_{r+1} & u_{r+2} & \cdots & u_m \end{bmatrix} \tag{3.23}$$

and we have the identity:

$$\mathbf{I} = \mathbf{U}_r\mathbf{U}_r^* + \mathbf{U}_{n-r}\mathbf{U}_{n-r}^* \tag{3.24}$$

since $\mathbf{I} = \mathbf{U}\mathbf{U}^* = \left(\begin{bmatrix}\mathbf{U}_r & 0\end{bmatrix} + \begin{bmatrix}0 & \mathbf{U}_{n-r}\end{bmatrix}\right)\left(\begin{bmatrix}\mathbf{U}_r & 0\end{bmatrix} + \begin{bmatrix}0 & \mathbf{U}_{n-r}\end{bmatrix}\right)^*$. Then, Eq.(3.21) can be written in terms of the singular matrices:

$$\begin{aligned}\left|\left|\phi(\xi^j) - \sum_{i=1}^{r}\langle\Phi_i, \phi(\xi^j)\rangle\Phi_i\right|\right|_2 &= \left|\left|\phi(\xi_j) - \mathbf{U}_r\mathbf{U}_r^*\phi(\xi^j)\right|\right|_2 \\ &= \left|\left|\mathbf{U}\mathbf{\Sigma}v_j^* - \mathbf{U}_r\mathbf{U}_r^*\mathbf{U}\mathbf{\Sigma}v_j^*\right|\right|_2 \\ &= \left|\left|\mathbf{U}_{n-r}\mathbf{U}_{n-r}^*\mathbf{U}\mathbf{\Sigma}v_j\right|\right|_2 \\ &\leq ||\mathbf{U}_{n-r}||_2\,||\Sigma_{n-r}||_2\left|\left|v_j^*\right|\right|_2 \\ &\leq \sigma_{r+1}\end{aligned} \tag{3.25}$$

since $||\mathbf{U}_{n-r}||_2 = 1$ and $\left|\left|v_j^*\right|\right|_2 = 1$. □

3.1.3 Range Finding Algorithm

Another way to generate a basis for POD is the range-finding algorithm proposed in [23]. In the range-finding algorithm, instead of Eq.(3.5), the error is calculated according to:

$$||(\mathbf{I} - \mathbf{Q}\mathbf{Q}^*)\mathbf{Y}|| \leq \epsilon. \tag{3.26}$$

This approach can be justified by considering Eq.(3.18). In Eq.(3.18), the POD-basis is given by the dominant left singular vectors of $\mathbf{Y}$. The left singular vectors form a basis for the range of $\mathbf{Y}$. Therefore, a good approximation of the range of $\mathbf{Y}$ will be a good candidate for $\mathbf{Q}$. If ϵ in Eq.(3.26) is small, then $\mathbf{Q}$ is a good approximation of the range of $\mathbf{Y}$. The range-finding algorithm to calculate the POD basis is summarized in Algorithm 4.

Algorithm 4 Range Finding Algorithm for the POD-Procedure [23]

Given an $n \times m$ matrix Y of snapshots, a tolerance ϵ, and an integer k, the following algorithm computes a POD basis $\mathbf{Q}$ such that Eq.(3.26) holds with probability at least 1-min $\{m, n\}10^{-k}$.

1. Draw standard Gaussian vectors $\omega^{(1)}$, ..., $\omega^{(k)}$ of length n.
2. For $i = 1, 2, \ldots, r$, compute $y^{(i)} = \mathbf{Y}\omega^{(i)}$.
3. $j = 0$
4. Set $\mathbf{Q} = 0_{n\times 0}$
5. **while** $\max\left\{\left|\left|y^{(j+1)}\right|\right|, \left|\left|y^{(j+2)}\right|\right|, \cdots, \left|\left|y^{(k+1)}\right|\right|\right\} \geq \epsilon/(10\sqrt{2/\pi})$
   (a) $j = j + 1$
   (b) Overwrite $y^{(j)}$ by $(\mathbf{I} - \mathbf{Q}^{(j-1)}(\mathbf{Q}^{(j-1)})^*)y^{(j)}$
   (c) $q^{(j)} = y^{(j)}/\left|\left|y^{(j)}\right|\right|$
   (d) $\mathbf{Q}^{(j)} = \begin{bmatrix}\mathbf{Q}^{(j-1)} & q^{(j)}\end{bmatrix}$
   (e) Draw a standard Gaussian vector $\omega^{(j+r)}$ of length n.
   (f) $y^{(j+r)} = (\mathbf{I} - \mathbf{Q}^{(j)}\mathbf{Q}^{(j)*})\mathbf{A}\omega^{(j+r)}$
   for
   - Overwrite $y^{(i)}$ by $y^{(i)} - q^{(i)}\langle q^{(i)}, y^{(i)}\rangle$

   end for
6. **end while**
7. $\mathbf{Q} = \mathbf{Q}^{(j)}$

Note that a better error bound for this form can be obtained using SVD given by:

$$\min_{\mathbf{Q}} ||(\mathbf{I} - \mathbf{Q}\mathbf{Q}^*)\mathbf{Y}|| = \sigma_{i+1} \tag{3.27}$$

where $\mathbf{Q} = \mathbf{U}_i$ and $\mathbf{U}_i$ is a partitioned matrix of $\mathbf{U} = [\mathbf{U}_i \ \mathbf{U}_{m-i}]$. This is known as the optimality of the singular value decomposition [25]. In other words, SVD is optimal and more accuracy in the sense of Eq.(3.26). However, SVD would be computationally more expansive than the range-finding algorithm. Therefore, the range-finding algorithm is preferred for large scale problems.

3.1.4 Implementation of Method of Snapshots by k-Least Squares

Assume that a solution to the SGEP can be written as a linear combination of the POD-basis given by:

$$\phi(\xi) \approx \sum_{i=1}^{r} \alpha(\xi)\Phi_i = \mathbf{Q}\alpha(\xi) \tag{3.28}$$

where $\mathbf{Q}$ is the POD-basis and $\alpha(\xi)$ is the vector of undetermined coefficients. Then, plug Eq.(3.28) into the SGEP:

$$\mathbf{L}(\xi)\phi(\xi) = \lambda(\xi)\mathbf{F}(\xi)\phi(\xi) \Rightarrow \mathbf{L}(\xi)\mathbf{Q}\alpha(\xi) = \lambda(\xi)\mathbf{L}(\xi)\mathbf{Q}\alpha(\xi) \tag{3.29}$$

where $\mathbf{L}$ and $\mathbf{F}$ are discretized matrices of dimension $n \times n$ and $\mathbf{L}(\xi)\mathbf{Q}$ and $\mathbf{F}(\xi)\mathbf{Q}$ are reduced matrices with dimension $n \times r$. Eq.(3.29) is called as overdetermined generalized eigenvalue problem. See [26] for its properties and how to solve this problem in general. Since $\mathbf{L}(\xi)\mathbf{Q}$ and $\mathbf{F}(\xi)\mathbf{Q}$ are no longer square, traditional approaches such as the power iterative method and Krylov-type subspace methods are not applicable to this problem. Therefore, Eq.(3.29) is transformed to an optimization problem to find a desired eigenpair. As discussed in Chapter 2, define a residual objective function:

$$\mathrm{R}(\alpha(\xi), \lambda(\xi)) = \begin{bmatrix} \mathbf{L}(\xi)\mathbf{Q} - \lambda(\xi)\mathbf{L}(\xi)\mathbf{Q} \\ \alpha(\xi)^T\alpha(\xi) - 1 \end{bmatrix} \tag{3.30}$$

where the second row of $\mathrm{R}(\alpha(\xi), \lambda(\xi))$ is equivalent to $\phi(\xi)^T\phi(\xi) = 1$ which is justified by:

$$\phi(\xi)^T\phi(\xi) \approx \alpha(\xi)^T\mathbf{Q}^T\mathbf{Q}\alpha(\xi) \approx \alpha(\xi)^T\alpha(\xi). \tag{3.31}$$

Note that the primary difference between Eq.(2.13) and Eq.(3.31) is that Eq.(2.13) becomes zero when $\lambda(\xi)$ and $\phi(\xi)$ are an eigenpair, but Eq.(3.31) is not guaranteed to become zero even when $\lambda(\xi)$ and $\alpha(\xi)$ are an eigenpair. This is due to the fact that $\mathbf{Q}\alpha(\xi)$ is an approximation of $\phi(\xi)$ and the error in the approximation prevents Eq.(3.31) from becoming zero. Therefore, Eq.(3.31) must be solved by finding a local minimum corresponding to an eigenpair:

$$\min_{\alpha(\xi),\lambda(\xi)} ||\mathrm{R}(\alpha(\xi),\lambda(\xi)||_2 \tag{3.32}$$

To find a local minimum, $\mathrm{R}(\alpha(\xi),\lambda(\xi))$ is expanded around the initial guess α_0 and λ_0 using a multivariate Taylor expansion:

$$\mathrm{R}(\alpha(\xi),\lambda(\xi)) = \mathrm{R}(\alpha_0,\lambda_0) + \mathbf{J}(\alpha_0,\lambda_0)\begin{bmatrix}\Delta\alpha\\ \Delta\lambda\end{bmatrix} + \cdots \tag{3.33}$$

where $\Delta\alpha = \alpha(\xi) - \alpha_0$ and $\Delta\lambda = \lambda(\xi) - \lambda_0$. It is important to note that $\mathbf{J}$ is no longer sparse. $\mathbf{J}$ is a dense matrix with dimension $n \times r$. The Jacobian matrix $\mathbf{J}(\alpha_0,\lambda_0)$ is given analytically by:

$$\mathbf{J}(\alpha(\xi),\lambda(\xi)) = \begin{bmatrix}\mathbf{L}(\xi)\mathbf{Q} - \lambda_0\mathbf{F}(\xi)\mathbf{Q} & -\mathbf{F}(\xi)\mathbf{Q}\alpha_0\\ 2\alpha_0 & 0\end{bmatrix} \tag{3.34}$$

Now, by neglecting the higher order terms, the Newton step is obtained by solving the least-square problem:

$$\begin{bmatrix}\Delta\alpha\\ \Delta\lambda\end{bmatrix} = -\mathbf{J}^{\dagger}(\alpha_0,\lambda_0)\mathrm{R}(\alpha_0,\lambda_0) \tag{3.35}$$

where $\mathbf{J}^{\dagger}(\alpha_0,\lambda_0)$ is the Moore-Penrose pseudo-inverse. Note that Eq.(3.35) can be solved by two different ways. The first approach is to factor $\mathbf{J}(\alpha_0,\lambda_0)$ by using the QR decomposition, $\mathbf{J} = \mathbf{Q_2}\mathbf{R_2}$ and the Newton step is then given by:

$$\begin{bmatrix}\Delta\alpha\\ \Delta\lambda\end{bmatrix} = -\mathbf{R}_2^{-1}\mathbf{Q}_2^{*}\mathrm{R}(\alpha_0,\lambda_0) \tag{3.36}$$

This approach is known to be numerically stable but computationally expensive. The second approach is to use the normal equation given by:

$$-\mathbf{J}^*(\alpha(\xi),\lambda(\xi))\mathrm{R}(\alpha_0,\lambda_0) = \mathbf{J}^*(\alpha(\xi),\lambda(\xi))\mathbf{J}(\alpha(\xi),\lambda(\xi))\begin{bmatrix}\Delta\alpha\\ \Delta\lambda\end{bmatrix}. \tag{3.37}$$

Then, the Newton step is obtained by:

$$\begin{bmatrix}\Delta\alpha\\ \Delta\lambda\end{bmatrix} = -\left(\mathbf{J}^*\mathbf{J}\right)^{-1}\mathbf{J}^*\mathrm{R}(\alpha_0,\lambda_0) \tag{3.38}$$

This approach is known to be computationally cheaper than the first approach but numerically unstable if the condition number of $\mathbf{J}$ is large. Since the Jacobian matrix is typically ill-conditioned[1], the second approach should be used with caution. Now, these approaches are denoted by k-LS method because the over-determined eigenvalue problem is solved by a least-square approach.

3.2 Discrete POD-Galerkin Method

The implementation of the method of snapshots described above may not have a unique solution. The overdetermined eigenvalue problem has been studied recently [26], and due to its formulation, there are concerns regarding its convergence to a correct answer. Therefore, instead of directly implementing the method of snapshots, the Galerkin projection is applied so that the dimension of the resulting system is square.

As a simplification, first consider a source driven problem, and then generalize it to an eigenvalue problem. A common stochastic linear problem can be written as:

$$\mathbf{A}(\xi)x(\xi) = s(\xi) \tag{3.39}$$

where $\mathbf{A}(\xi)$ is a $n \times n$ linear operator, $s(\xi)$ is the stochastic source term, and $x(\xi)$ is the solution to the system. Using the POD procedure described in the previous section, a solution to a stochastic linear equation can be written as a unique combination of

[1]This is the reason why a preconditioner is needed for JFNK methods.

solutions generated from realizations of $x(\xi)$ described as:

$$x(\xi) \approx \sum_{r=1}^{m} \alpha(\xi)\Phi_i = \mathbf{Q}\alpha(\xi). \tag{3.40}$$

Then, plugging Eq.(3.40) to Eq.(3.39), a stochastic linear equation can be written in terms of the undetermined coefficients $\alpha(\xi)$:

$$\mathbf{A}(\xi)\mathbf{Q}\alpha(\xi) = s(\xi) \tag{3.41}$$

where $\mathbf{A}(\xi)\mathbf{Q}$ is $n \times r$. In order to avoid having to solve a over-determined system, we want to reduce the system to be $r \times r$. This can be achieved by projecting Eq.(3.41) onto 1. $\mathbf{Q}$, 2. the range of $\mathbf{A}$, or 3. the range of $s(\xi)$. In the discrete Galerkin projection, one common approach is that Eq.(3.41) is projected onto $\mathbf{\Phi}_i$ given by:

$$\langle \Phi_i, A(\xi)x(\xi)\rangle = \langle \Phi_i, s(\xi)\rangle \text{ for } i = 1, \cdots, r. \tag{3.42}$$

In matrix form, Eq.(3.42) can be written as:

$$\mathbf{Q}^*\mathbf{A}(\xi)\mathbf{Q}\alpha(\xi) = \mathbf{Q}^* s(\xi). \tag{3.43}$$

Now, $\mathbf{A}$ is reduced to $\mathbf{Q}^*\mathbf{A}(\xi)\mathbf{Q}$ which is $r \times r$. Note that $\mathbf{A}$ is a sparse matrix and $\mathbf{Q}^*\mathbf{A}(\xi)\mathbf{Q}$ is a *dense* matrix. Likewise, it is not surprise that the same procedure can be applied to the eigenvalue problem in Eq.(3.29). Projecting Eq.(3.29) onto $\mathbf{Q}$:

$$\mathbf{Q}^*\mathbf{L}(\xi)\mathbf{Q}\alpha(\xi) = \lambda\mathbf{Q}^*\mathbf{F}(\xi)\mathbf{Q}\alpha(\xi) \tag{3.44}$$

which is a reduced-order eigenvalue problem. It is important to discuss when this approach will work and will not work. Unfortunately, this approach is not guaranteed to preserve the eigenvalue of interest. To show this, denote Eq.(3.44) as:

$$\mathbf{Q}^*\mathbf{L}(\xi)\mathbf{Q}\alpha(\xi) = \lambda_R\mathbf{Q}^*\mathbf{F}(\xi)\mathbf{Q}\alpha(\xi) \tag{3.45}$$

where λ_R is a right eigenvalue. Now, consider an adjoint problem to a reduced order

eigenvalue problem that is obtained by the transposing $\mathbf{Q}^*\mathbf{L}\mathbf{Q}$ and $\mathbf{Q}^*\mathbf{F}\mathbf{Q}$:

$$\mathbf{Q}^*\mathbf{L}^*(\xi)\mathbf{Q}\alpha^*(\xi) = \lambda_L\mathbf{Q}^*\mathbf{F}^*(\xi)\mathbf{Q}\alpha^*(\xi) \tag{3.46}$$

where $\lambda_L(\xi)$ is a left eigenvalue and $\alpha^*(\xi)$ is the adjoint solution. Now, one can prove that $\lambda_R = \lambda_L$ must hold when $\langle\alpha^*(\xi), \alpha(\xi)\rangle \neq 0$.

Without loss of generality, since $\mathbf{Q}^*\mathbf{L}(\xi)\mathbf{Q}$ is non-singular, Eq.(3.44)-3.46 can be written in the form:

$$\mathbf{A}(\xi)\alpha(\xi) = k_R(\xi)\alpha(\xi) \tag{3.47}$$

where $\mathbf{A}(\xi) = (\mathbf{Q}^*\mathbf{L}(\xi)\mathbf{Q})^{-1}\mathbf{Q}^*\mathbf{F}(\xi)\mathbf{Q}$ and $k_R(\xi) = 1/\lambda_R(\xi)$. Note that $\mathbf{A}(\xi)$ is not always non-singular depending on the structure of $\mathbf{F}(\xi)$. Now, we can apply the properties of an eigenvalue problem of the form, Eq.(3.47).

Theorem 1. *Suppose* $\mathbf{A}(\boldsymbol{\xi})$ *is a square matrix and* k *is an eigenvalue of* $\mathbf{A}(\boldsymbol{\xi})$*. Then* k *is also an eigenvalue of the matrix* $\mathbf{A}^*(\boldsymbol{\xi})$[2]*.*

Proof. Assume that $q(k)$ is the characteristic polynomial of $A(\xi)$. Then,

$$\begin{aligned} q_{\mathbf{A}(\xi)}(k) &= \det(\mathbf{A}(\xi) - k\mathbf{I}) \\ &= \det((\mathbf{A}(\xi) - k\mathbf{I})^*) \\ &= \det(\mathbf{A}^*(\xi) - k\mathbf{I}) \\ &= q_{\mathbf{A}^*(\xi)}(k) \end{aligned}$$

which implies that $\mathbf{A}(\xi)$ and $\mathbf{A}^*(\xi)$ have the same characteristic polynomial. Then, when k is an eigenvalue of $\mathbf{A}(\xi)$, $q_{\mathbf{A}(\boldsymbol{\xi})}(k) = 0$. Equivalently, $q_{\mathbf{A}^*(\xi)}(k) = 0$ which implies $\mathbf{A}(\xi)$ and $\mathbf{A}^*(\xi)$ have the same eigenvalues. □

From Theorem 1, an adjoint equation of a reduced order model must have the same eigenvalues. In other words, if an eigenvalue of the adjoint system cannot be calculated accurately, and then the reduced-order system should get the same and inaccurate eigenvalue as the adjoint system shown in the above theorem. From Eq.(3.46), the adjoint solution obtained should be of the form:

$$\phi^*(\xi) \approx \mathbf{Q}\alpha^*(\xi) \tag{3.48}$$

[2]This proof can be found in most of elementary linear algebra books

and Eq.(3.48) holds if and only if

$$\phi^*(\xi) \in \text{span}\,\{\Phi_1, \cdots, \Phi_r\} \tag{3.49}$$

which is in general *not* true. Since Φ_i spans the forward solutions and there is no relationship between a forward solution and an adjoint solution, Φ_i typically does not span $\phi^*(\xi)$. Therefore, an adjoint system of a reduced order problem cannot be calculated accurately resulting in an inaccurate eigenvalue in an adjoint problem and forward problem. Hence the discrete POD-Galerkin method is in general not applicable to an eigenvalue problem and cannot be trusted unless $\mathbf{Q}$ spans the adjoint solution.

3.3 Mixed Forward-Adjoint Basis Approach

From the discussion above, it is obvious that the discrete POD-Galerkin method may fail because the SGEP is projected onto an inappropriate space which does not span an adjoint solution. Therefore, the SGEP must be projected onto a space which spans an adjoint solution. From Theorem 1, one can easily find that for $\langle \alpha^*, \alpha \rangle \neq 0$, when $\mathbf{Q}_L$ spans an adjoint solution:

$$\begin{aligned} \lambda_R = \lambda_L &= \frac{\phi^*(\xi)\mathbf{L}(\xi)\phi(\xi)}{\phi^*(\xi)\mathbf{F}(\xi)\phi(\xi)} \\ &\approx \frac{\alpha^*(\xi)\mathbf{Q}_L^*\mathbf{L}(\xi)\mathbf{Q}_R\alpha(\xi)}{\alpha^*(\xi)\mathbf{Q}_L^*\mathbf{F}(\xi)\mathbf{Q}_R\alpha(\xi)} \end{aligned} \tag{3.50}$$

Then, λ_R and λ_L are eigenvalues of the matrix pencil $(\mathbf{Q}_L^*\mathbf{L}(\xi)\mathbf{Q}_R, \mathbf{Q}_L^*\mathbf{F}(\xi)\mathbf{Q}_R)$. Therefore, a reduced eigenvalue problem given by:

$$\mathbf{Q}_L^*\mathbf{L}(\xi)\mathbf{Q}_R\alpha(\xi) = \lambda\mathbf{Q}_L^*\mathbf{F}(\xi)\mathbf{Q}_R\alpha(\xi) \tag{3.51}$$

which preserves an eigenvalue of the original problem as long as $\mathbf{Q}_L$ spans the adjoint solution. Since $\mathbf{Q}_L$ forms a basis for or an adjoint solution lies in the span of an adjoint solution, it is intuitive that orthogonal matrix $\mathbf{Q}_L$ should be generated by the realizations of adjoint solutions with randomly perturbed cross-section values. To do this, add adjoint

solutions into Eq.(3.6) and weight it by γ:

$$\phi^*(\xi) \in \text{span}\left\{\gamma\phi(\xi^1) + (1-\gamma)^*\phi(\xi^1), \cdots, \phi(\xi^m) + (1-\gamma)\phi^*(\xi^m)\right\} \tag{3.52}$$

where $\gamma \in [0, 1)$. It is obvious that unless γ is close to 1. Eq.(3.52) spans an adjoint solution. Now, form $\tilde{\mathbf{Y}} = \left[\gamma\phi(\xi^1) + (1-\gamma)\phi^*(\xi^1)) \quad \cdots \quad \phi(\xi^m) + (1-\gamma)\phi^*(\xi^m)\right]$. Then, the left orthogonal basis can be obtained by:

$$\tilde{\mathbf{Y}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^* \tag{3.53}$$

where $\mathbf{Q}_L$ is equal to $\tilde{\mathbf{U}}$. Then, Eq.(3.29) can be reduced by projecting onto $\mathbf{Q}_L$:

$$\mathbf{Q}_L\mathbf{L}(\xi)\mathbf{Q}_R\alpha(\xi) = \lambda\mathbf{Q}_L\mathbf{F}(\xi)\mathbf{Q}_R\alpha(\xi). \tag{3.54}$$

Unlike the overdetermined eigenvalue problem, Eq.(3.54) is guaranteed to have r solutions. It also preserves the eigenvalue of interest. However, it introduces $r-1$ new eigenvalues which does not exist in the original problem and which could be larger than the largest eigenvalue in the original problem. Therefore, the power method is no longer applicable in a reduced order problem. Hence, different approaches such as the Newton Method should be used for solving a reduced order problem.

It is important to note that extra attention must be paid to the initial guess of $\alpha(\xi)$ because of $\mathbf{Q}_R$. Depending on how $\mathbf{Q}_R$ is computed, $\mathbf{Q}_R$ can be a flipped image of $\phi(\xi^j)$. In such a case, the initial guess also must be flipped by multiplying -1. Otherwise, the solution converges to a different eigenpair.

Other Benefits of ROM

In this chapter, state-level ROM techniques based on POD-procedure are discussed. ROM techniques are introduced to reduce the size of the problem dimension. However, there are also other benefits. Consider a large scale problem which requires distributed memory. In such a problem, one needs an efficient preconditior to solve a linear system or an eigenvalue problem. To computer such a preconditioner, one needs to pay extra attention to distributed algorithms in order to minimize the amount of memory required to store such a preconditioner and appropriate iterative schemes to minimize

the computational time. To perform such a task, one needs to be highly experienced with distributed computing and to have strong background in linear algebra. On the other hand, ROM techniques free researchers from such tasks. Once matrix operators are reduced using POD-basis, then one no longer needs to consider any preconditioner and distributed computing since the dimension of the problem is small compared with the original system and a personal computer is sufficient to solve a reduced order system. In addition, a reduced order system is no longer sparse, and therefore only direct solvers are required which is far simpler than the iterative methods.

Chapter 4

generalized Polynomial Chaos for Uncertainty Quantification

Uncertainty Quantification (UQ) is a rapidly growing discipline in reactor simulation due to its importance in the safety and economics of the associated nuclear reactor system. UQ is the process of determining uncertainties in the results propagated from uncertainties in the input parameters such as uncertainties in the cross-section data input to the neutronics models. To perform UQ, sampling methods such as Monte Carlo method (MC) are commonly utilized for UQ. MC is a non-intrusive approach. In addition, another desirable feature of MC is that the curse of dimensionality does not affect its convergence rate. Regardless the number of stochastic variables, MC will converge with $\mathcal{O}(n^{-1/2})$. This property is well-suited for an engineering model with hundreds and thousands of stochastic variables in the input parameter space. However, $\mathcal{O}(n^{-1/2})$ is slow and the computational cost is typically large for reactor simulation. Therefore, in order to accelerate the convergence rate in MC, Quasi-Monte Carlo method (QMC) is widely used for a model with few stochastic variables [27]. QMC typically exhibits a faster convergence rate than MC. However, the convergence rate of QMC is affected by the curse of dimensionality and typically the convergence rate decreases as the number of stochastic variables increases.

Recently, two approaches have emerged as alternatives to MC: the sparse grid approximation (SG), and the generalized polynomial chaos (gPC). In SG methods, the response is approximated by a linear combination of hierarchical hat basis functions built on the Smolyak-type sparse grid [17]. SG approximation also suffers from the curse of dimension-

ality. However, the dimension-adaptive sparse grid approximation has been proposed to overcome the curse of dimensionality and the dimension-adaptive SG shows an excellent convergence rate [28].

In gPC, a spectral decomposition approach is used to expand the response function in terms of an orthogonal function basis [29]. gPC can be implemented intrusively and non-intrusively. In the intrusive gPC implementation, the stochastic model equations are converted into a system of deterministic equation by Galerkin projection to solve not only for the reference solution but also for all the coefficients of the orthogonal basis functions. Then, gPC can calculate the higher order moments of the response relatively easily. Although this approach provides detailed information about response uncertainties, it has several drawbacks. First, the Galerkin projection is computationally expensive for a model with complicated geometry and many stochastic variables. This is the case for reactor simulation. In addition, the number of orthogonal basis vectors exponentially grows with the number of stochastic variables. Intrusive gPC has been applied for an eigenvalue problem and the application to neutronics problem has been studied in [30]. In non-intrusive gPC implementations, the response is sampled and a Vandermonde-like system of equations is constructed to find the coefficients for the orthogonal basis functions. Both intrusive and non-intrusive gPC face the curse of dimensionality. However, the sparse approximation has been proposed for non-intrusive gPC.

Orthogonal Polynomial Basis

In gPC, a stochastic response is approximated by orthogonal basis functions. Assume that the input parameters have s stochastic variables represented by $\xi = [\, \xi_1 \cdots \xi_i \cdots \xi_s \,]^T$ where each ξ_i has its own probability density function where $\xi \in \mathcal{Y}$ and $\mathcal{Y}$ is stochastic space. Then, response is represented by the chaos expansion given by:

$$\mathrm{R}(\xi) = \sum_{i=0}^{\infty} a_i \Psi_i(\xi) \tag{4.1}$$

where a_i is the i-th polynomial chaos coefficient and $\Psi_i(\xi)$ is the i-th multivariate orthogonal basis function. Note that in a case of Legendre polynomials, $\Psi_i(\xi)$ can be expressed

as:

$$\begin{aligned}
\Psi_0(\xi) &= \mathrm{P}_0(\xi_1) = \cdots = \mathrm{P}_0(\xi_s) \\
\Psi_1(\xi) &= \mathrm{P}_1(\xi_1) \\
&\vdots \\
\Psi_{s+1}(\xi) &= \mathrm{P}_1(\xi_s) \\
\Psi_{s+2}(\xi) &= \mathrm{P}_2(\xi_1) \\
\Psi_{s+3}(\xi) &= \mathrm{P}_1(\xi_1)\mathrm{P}_1(\xi_2) \\
&\vdots
\end{aligned}$$

and $\mathrm{P}_i(\xi_j)$ is a univariate i-th order orthogonal polynomial of ξ_j. Note that it has an orthogonal relationship given by:

$$\int d\xi_k \mathrm{P}_i(\xi_k)\mathrm{P}_j(\xi_k)w(\xi_k) = \delta_{ij} \int d\xi_k \mathrm{P}_i^2(\xi_k)w(\xi_k) \tag{4.2}$$

where $w(\xi_k)$ is a weighting function. Using this orthogonality, the orthogonal property for the multivariate function is given by:

$$\int d\xi \Psi_i(\xi)\Psi_j(\xi)w(\xi) = \delta_{ij} \int d\xi \Psi_i^2(\xi)w(\xi) \tag{4.3}$$

where $w(\xi) = \prod_{k=1}^{s} w(\xi_k)$. The type of the orthogonal function $\mathrm{P}_j(\xi_k)$ is determined by the type of a continuous probability distribution of ξ_k. This is called the Askey scheme [31]. Table 4.1 shows the Askey scheme for the continuous hyergeometric polynomials.

Table 4.1: Askey scheme of continuous hypergeometric polynomials

Distribution	Density function	Polynomial P	Weight function
Normal	$\frac{1}{2\pi}e^{-\frac{x^2}{2}}$	Hermite	$e^{-\frac{x^2}{2}}$
Uniform	$\frac{1}{2}$	Legendre	1
Beta	$\frac{(1-x)^\alpha(1+x)^\beta}{2^{\alpha+\beta+1}B(\alpha+1,\beta+1)}$	Jacobi	$(1-x)^\alpha(1+x)^\beta$
Exponential	e^{-x}	Laguerre	e^{-x}
Gamma	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$	Generalized Laguerre	$x^\alpha e^{-x}$

In practice, Eq.(4.1) is truncated at a finite expansion order M:

$$\mathrm{R}(\xi) = \sum_{i=0}^{M} a_i \Psi_i(\xi). \tag{4.4}$$

There are multiple approaches how the polynomial chaos expansion is truncated. The most common approach is the total order expansion where the order of expansion is up to a fixed order. In this case, the number of term M with an expansion of total order p involving s stochastic variables is given by:

$$M + 1 = \frac{(s+p)!}{s!p!}. \tag{4.5}$$

It is obvious that the number of polynomial terms exponentially increases with s and p. This is called the curse of dimensionality. In a stochastic partial differential equation, this problem is quite common and the computational cost can get easily intractable in most cases.

Once the chaos expansion has been obtained, the mean and variance of the response R can be obtained. The mean value of R can be obtained by:

$$\mu(\mathrm{R}(\xi)) = \frac{1}{V} \sum_{i=0}^{M} a_i \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_i(\xi) \tag{4.6}$$

where V is the volume of the stochastic space, $\mathcal{Y}$. In addition, the variance of $\mathrm{R}(\xi)$ is obtained by:

$$\begin{aligned}
\mathrm{Var}(\mathrm{R}(\xi)) &= \mu(\mathrm{R}(\xi)^2) - \mu(\mathrm{R}(\xi))^2 \\
&= \frac{1}{V} \int_{\mathcal{Y}} \mathrm{d}\xi \ \left(\sum_{i=0}^{M} a_i \Psi_i(\xi) \right)^2 - \left(\frac{1}{V} \sum_{i=0}^{M} a_i \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_i(\xi) \right)^2 \\
&= \frac{1}{V} \sum_{i=0}^{M} \sum_{j=0}^{M} a_i a_j \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_i(\xi) \Psi_j(\xi) - \frac{1}{V^2} \sum_{i=0}^{M} \sum_{j=0}^{M} a_i a_j \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_i(\xi) \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_j(\xi).
\end{aligned} \tag{4.7}$$

It is important to note that when the multivariate orthogonal basis function is a Legendre

polynomial, the mean is simplified to:

$$\begin{aligned}\mu(\mathrm{R}(\xi)) &= \frac{1}{V}\sum_{i=0}^{M} a_i \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_i(\xi) \\ &= \frac{1}{V}\sum_{i=0}^{M} a_i \int_{\mathcal{Y}} \mathrm{d}\xi \ \frac{\Psi_0(\xi)}{\Psi_0(\xi)}\Psi_i(\xi) \\ &= \frac{1}{V\Psi_0(\xi)}\sum_{i=0}^{M} a_i \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_0(\xi)\Psi_i(\xi) \\ &= \frac{a_0}{V}V = a_0 \end{aligned} \tag{4.8}$$

since $\Psi_0(\xi) = 1$. Also, the variance is obtained from:

$$\begin{aligned}\mathrm{Var}(\mathrm{R}(\xi)) &= \mu(\mathrm{R}(\xi)^2) - \mu(\mathrm{R}(\xi))^2 \\ &= \frac{1}{V}\int_{\mathcal{Y}} \mathrm{d}\xi \ \left(\sum_{i=0}^{M} a_i\Psi_i(\xi)\right)^2 - a_0^2 \\ &= \frac{1}{V}\sum_{i=0}^{M}\sum_{j=0}^{M} a_i a_j \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_i(\xi)\Psi_j(\xi) - a_0^2 \\ &= \frac{1}{V}\sum_{i=0}^{M} a_i^2 \int_{\mathcal{Y}} \mathrm{d}\xi \ (\Psi_i(\xi))^2 - a_0^2 \\ &= \frac{1}{V}\sum_{i=1}^{M} a_i^2 \langle \Psi_i(\xi), \Psi_i(\xi)\rangle. \end{aligned} \tag{4.9}$$

Of note is that the last inner product is easily calculated by using the orthogonal property of the Legendre polynomial:

$$\int_{-1}^{1} \mathrm{d}x \mathrm{P}_m(x)\mathrm{P}_n(x) = \frac{2}{2n+1}\delta_{mn}. \tag{4.10}$$

4.1 Intrusive Polynomial Chaos

We will first assume that the multi-group diffusion equation Eq.(2.5) has uncertainties in the cross-section values. The uncertainties propagate to uncertainties in the flux distribution and the responses of interest. To describe these uncertainties, the stochastic

variable ξ, which is a vector, is introduced into Eq.(2.5) and integrated over the energy group:

$$
\begin{aligned}
&-\nabla\cdot D_g(r,\xi)\nabla\phi_g(r,\xi)+\Sigma_t^g(r,\xi)\phi_g(r,\xi)\\
&\quad=\sum_{g'=1}^{G}\Sigma_s^{g'\to g}(r,\xi)\phi_{g'}(r,\xi)+\frac{\chi_g(r,\xi)}{k(\xi)}\sum_{g'=1}^{G}\nu_g(r,\xi)\Sigma_f^{g'}(r,\xi)\phi_{g'}(r,\xi),\quad g=1,\cdots G
\end{aligned}
\tag{4.11}
$$

where $r\in\Omega$ and $\xi\in\mathcal{Y}$. In most cases, an analytical solution to this stochastic partial differential equation is not known and extremely hard to solve for large problems like an assembly model due to large number of stochastic variables.

4.1.1 Source-Driven Problem

To simplify the problem, consider first the fission source-driven problem of Eq.(4.11) denoted as:

$$
-\nabla\cdot D_g(r,\xi)\nabla\phi_g(r,\xi)+\Sigma_t^g(r,\xi)\phi_g(r,\xi)-\sum_{g'=1}^{G}\Sigma_s^{g'\to g}(r,\xi)\phi_{g'}(r,\xi)=s_g(r,\xi),\quad g=1,\cdots G
\tag{4.12}
$$

where the fission source term is given by:

$$
s_g(r,\xi)=\frac{\chi_g(r,\xi)}{k^{ref}}\sum_{g'=1}^{G}\nu_g(r,\xi)\Sigma_f^{g'}(r,\xi)\phi_{g'}^{ref}(r)
\tag{4.13}
$$

where k^{ref} and $\phi_{g'}^{ref}(r)$ are a solution to a reference case. To solve Eq.(4.12), the scalar flux is expanded by the generalized polynomial chaos expansion described by:

$$
\phi_g(r,\xi)=\sum_{i=0}^{\infty}f_{g,i}(r)\Psi_i(\xi)
\tag{4.14}
$$

where $f_{g,i}(r)$ are the undetermined coefficients of the multivariate polynomial $\Psi_i(\xi)$. Then, assume that each $f_{g,i}(r)$ can be decomposed by a set of orthogonal functions

$\Phi_j(r)$:

$$f_{g,i}(r) = \sum_{j=1}^{\infty} \alpha_{g,ij} \Phi_j(r) \tag{4.15}$$

where $\Phi_j(r)$ satisfies the orthogonal property given by:

$$\langle \Phi_j(r), \Phi_k(r) \rangle = \delta_{ij} \langle \Phi_j(r), \Phi_j(r) \rangle. \tag{4.16}$$

Now, combining Eq.(4.14) and Eq.(4.15), the scalar flux can be represented as:

$$\phi_g(r, \xi) = \sum_{i=0}^{\infty} \sum_{j=1}^{\infty} \alpha_{g,ij} \Phi_j(r) \Psi_i(\xi). \tag{4.17}$$

Note that $\Phi_j(r)$ can be chosen from any family of orthogonal functions. Therefore, without loss of generality, $\Phi_j(r)$ can be assumed to be generated by the POD procedure described in the Chapter 3. To solve for α_{ij}, first of all Eq.(4.17) is truncated to the finite number of terms denoted as:

$$\phi_g(r, \xi) \approx \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g,ij} \Phi_j(r) \Psi_i(\xi). \tag{4.18}$$

Then, substitute Eq.(4.18) into Eq.(4.12):

$$\begin{aligned} -\nabla \cdot D_g(r, \xi) \nabla \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g,ij} \Phi_j(r) \Psi_i(\xi) + \Sigma_t^g(r, \xi) \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g,ij} \Phi_j(r) \Psi_i(\xi) \\ - \sum_{g'=1}^{G} \Sigma_s^{g' \to g}(r, \xi) \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g',ij} \Phi_j(r) \Psi_i(\xi) = s_g(r, \xi), \quad g = 1, \cdots G \end{aligned} \tag{4.19}$$

where the only unknowns are $\alpha_{g,ij}$ and $\alpha_{g',ij}$. To determine these unknowns, the Galerkin projection (more broadly the method of weighted residuals) is applied to Eq.(4.19) by projecting Eq.(4.19) onto $\mathrm{P}_l(\xi)$ and $\Phi_m(r)$. That is, multiply $\Psi_l(\xi)$ and $\Phi_m(r)$ to Eq.(4.19)

and integrate it over the stochastic space and spatial domain given by:

$$
\begin{aligned}
-\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\nabla\cdot D_g(r,\xi)\nabla\Phi_j(r)\Psi_i(\xi)& \\
+\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_t^g(r,\xi)\Phi_j(r)\Psi_i(\xi)& \\
-\sum_{i=0}^{M}\sum_{j=1}^{r}\sum_{g'=1}^{G}\alpha_{g',ij}\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_s^{g'\to g}(r,\xi)\Phi_j(r)\Psi_i(\xi)& \\
=\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)s_g(r,\xi),\quad g=1,\cdots G&
\end{aligned}
\tag{4.20}
$$

where all terms are now deterministic coefficients except $\alpha_{g,ij}$ and $\alpha_{g',ij}$. To simplify Eq.(4.20), denote that:

$$
\begin{aligned}
D_{g,ijlm}&=\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\nabla\cdot D_g(r,\xi)\nabla\Phi_j(r)\Psi_i(\xi)\\
\Sigma_{g,ijlm}^{t}&=\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_t^g(r,\xi)\Phi_j(r)\Psi_i(\xi)\\
\Sigma_{g'\to g,ijlm}^{s}&=\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_s^{g'\to g}(r,\xi)\Phi_j(r)\Psi_i(\xi)\\
s_{g,lm}&=\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)s_g(r,\xi)
\end{aligned}
\tag{4.21}
$$

and Eq.(4.20) can be written as:

$$
\begin{aligned}
-\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}D_{g,ijlm}+\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}\Sigma_{g,ijlm}^{t}& \\
-\sum_{i=0}^{M}\sum_{j=1}^{r}\sum_{g'=1}^{G}\alpha_{g',ij}\Sigma_{g'\to g,ijlm}^{s}=s_{g,lm},\quad g=1,\cdots G.&
\end{aligned}
\tag{4.22}
$$

Now, Eq.(4.22) can be written as the dense matrix equation for $\alpha_{g,ij}$ and $\alpha_{g',ij}$ and is ready to be solved.

4.1.2 k-Eigenvalue Problem

Different from the fission source-driven problem, the k-eigenvalue problem is more complicated to solve, yet the formulation is almost the same. Assume that the reciprocal of k-eigenvalue can be expanded by the chaos expansion given by:

$$\frac{1}{k(\xi)} = \lambda(\xi) = \sum_{i=0}^{\infty} \lambda_i \Psi_i(\xi) \tag{4.23}$$

where λ_i are the undetermined coefficients of the chaos expansion for $\lambda(\xi)$. Of note is that the expansion does not necessarily need to be the reciprocal of k. It could be the case that:

$$k(\xi) = \sum_{i=0}^{\infty} k_i \Psi_i(\xi) \tag{4.24}$$

where k_i are undetermined coefficients of the chaos expansion for $k(\xi)$. Both equations can be used for the intrusive polynomial chaos formulation of for the diffusion equation, but the resulting system of equations are slightly different. In this chapter, only Eq.(4.23) is considered, but the same argument is valid for Eq.(4.24).

Similar to the source problem, the scalar flux is expanded using the chaos expansion and the POD basis. First of all, Eq.(4.23) is truncated to a finite number of terms:

$$\frac{1}{k(\xi)} = \lambda(\xi) = \sum_{i=0}^{M} \lambda_i \Psi_i(\xi) \tag{4.25}$$

Then, substitute Eq.(4.18) and Eq.(4.25) into Eq.(4.11):

$$\begin{aligned} &- \nabla \cdot D_g(r,\xi) \nabla \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g,ij} \Phi_j(r) \Psi_i(\xi) \\ &\quad + \Sigma_t^g(r,\xi) \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g,ij} \Phi_j(r) \Psi_i(\xi) - \sum_{g'=1}^{G} \Sigma_s^{g' \to g}(r,\xi) \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g',ij} \Phi_j(r) \Psi_i(\xi) \\ &= \chi_g(r,\xi) \sum_{k=0}^{M} \lambda_k \Psi_k(\xi) \sum_{g'=1}^{G} \nu_g(r,\xi) \Sigma_f^{g'}(r,\xi) \sum_{i=0}^{M} \sum_{j=1}^{r} \alpha_{g',ij} \Phi_j(r) \Psi_i(\xi), \quad g = 1, \cdots G \end{aligned} \tag{4.26}$$

Taking the Galerkin projection onto the basis yields the following deterministic equation:

$$
\begin{aligned}
&-\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\nabla\cdot D_g(r,\xi)\nabla\Phi_j(r)\Psi_i(\xi)\\
&+\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_t^g(r,\xi)\Phi_j(r)\Psi_i(\xi)\\
&-\sum_{i=0}^{M}\sum_{j=1}^{r}\sum_{g'=1}^{G}\alpha_{g',ij}\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_s^{g'\to g}(r,\xi)\Phi_j(r)\Psi_i(\xi)\\
&=\sum_{i=0}^{M}\sum_{j=1}^{r}\sum_{k=0}^{M}\sum_{g'=1}^{G}\alpha_{g',ij}\lambda_k\\
&\quad\int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\chi_g(r,\xi)\Psi_k(\xi)\nu_g(r,\xi)\Sigma_f^{g'}(r,\xi)\Phi_j(r)\Psi_i(\xi),\quad g=1,\cdots G. \quad (4.27)
\end{aligned}
$$

To simplify the above equation, denote as the following:

$$
\begin{aligned}
D_{g,ijlm} &= \int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\nabla\cdot D_g(r,\xi)\nabla\Phi_j(r)\Psi_i(\xi)\\
\Sigma^t_{g,ijlm} &= \int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_t^g(r,\xi)\Phi_j(r)\Psi_i(\xi)\\
\Sigma^s_{g'\to g,ijlm} &= \int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\Sigma_s^{g'\to g}(r,\xi)\Phi_j(r)\Psi_i(\xi)\\
s_{g',ijklm} &= \int_{\mathcal{Y}}\mathrm{d}\xi\int_{\Omega}\mathrm{d}r\ \Psi_l(\xi)\Phi_m(r)\chi_g(r,\xi)\Psi_k(\xi)\nu_g(r,\xi)\Sigma_f^{g'}(r,\xi)\Phi_j(r)\Psi_i(\xi) \quad (4.28)
\end{aligned}
$$

and Eq.(4.27) simplifies to:

$$
\begin{aligned}
&-\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}D_{g,ijlm}+\sum_{i=0}^{M}\sum_{j=1}^{r}\alpha_{g,ij}\Sigma^t_{g,ijlm}\\
&\qquad-\sum_{i=0}^{M}\sum_{j=1}^{r}\sum_{g'=1}^{G}\Sigma^s_{g'\to g,ijlm}=\sum_{i=0}^{M}\sum_{j=1}^{r}\sum_{k=0}^{M}\sum_{g'=1}^{G}\alpha_{g',ij}\lambda_k s_{g',ijklm},\quad g=1,\cdots G \quad (4.29)
\end{aligned}
$$

where the number of equation is $gr(M+1)$ and the number of unknowns are $(gr+1)(M+1)$. Therefore, Eq.(4.29) is under-determined system. An additional $M+1$ equations can

be obtained by considering the normalization constraint:

$$\sum_{i=1}^{G} \int_{\Omega} \mathrm{d}r \ \phi_g(r,\xi)\phi_g(r,\xi) = 1. \tag{4.30}$$

Then, the above equation can be transformed by the POD-gPC expansion described in Eq.(4.18):

$$\sum_{i=1}^{G}\sum_{j=0}^{M}\sum_{k=1}^{r}\sum_{l=0}^{M}\sum_{m=1}^{r} \alpha_{g,jk}\alpha_{g,lm}\Psi_k(\xi)\Psi_m(\xi) \int_{\Omega} \mathrm{d}r \ \Phi_j(r), \Phi_l(r) = 1 \tag{4.31}$$

since $\Phi_i(\xi)$ is an orthogonal function:

$$\sum_{i=1}^{G}\sum_{j=0}^{M}\sum_{k=1}^{r}\sum_{m=0}^{M} \alpha_{g,jk}\alpha_{g,jm}\Psi_k(\xi)\Psi_m(\xi) = 1 \tag{4.32}$$

and taking a Galerkin projection onto the stochastic basis results in the following $(M+1)$ equations,

$$\sum_{i=1}^{G}\sum_{j=0}^{M}\sum_{k=1}^{r}\sum_{m=0}^{M} \alpha_{g,jk}\alpha_{g,jm} \int_{\mathcal{Y}} \mathrm{d}\xi \ \Psi_k(\xi)\Psi_m(\xi)\Psi_n(\xi) = \delta_{n0} \quad \text{for } n = 0, \ldots, M \tag{4.33}$$

To solve this system of equations, Newton's Method discussed in the Chapter 2 can be applied.

Limitation of the Intrusive Approach

Even though intrusive gPC is mathematically possible and optimal in L_2 norm, it is common that the method is computationally infeasible. Consider Eq.(4.28) as an example. To obtain $s_{g',ijklm}$, a symbolic or numerical integration over ξ and r must be performed. However, r has 3 dimensions and ξ has s dimensions. When the geometry is complicated such as with an assembly configuration, the numerical integration on r is very expensive because most coefficients such as $D(r,\xi)$ and $\Sigma(r,\xi)$ are discontinuous functions with respect to r. In addition, numerical integration over ξ is more expensive since the number of stochastic variables is s, which could be more than 100 even for a simple configuration. These difficulties with numerical integration make intrusive gPC intractable for reactor

simulation and only applicable to a small and simple problem with a few stochastic variables.

4.2 Non-Intrusive Polynomial Chaos

In most cases, intrusive gPC is not applicable for reactor simulation mainly because its difficulties and computational costs in numerical integration. An alternative approach is to find $\alpha_{g,ij}$ and $k_i(\xi)$ with a non-intrusive method broadly known as non-intrusive generalized polynomial chaos. In non-intrusive gPC, the k-eigenvalue and scalar fluxes are sampled with random perturbation of cross-section values and $\alpha_{g,ij}$ and $k_i(\xi)$ are obtained by solving the Vandermonde-like matrix.

In non-intrusive gPC, no modification in source code is required. An existing code treated is as a black-box that is evaluated many times to sample solutions and a matrix equation describing a relation between stochastic inputs and outputs is constructed. Then, a matrix equation is solved to recover the relation. Since no modification to the existing code is required, the non-intrusive approach is preferable in most cases and this approach can be easily implemented.

Assume that a black-box accepts perturbed cross-section values as input parameters and generates spatially-discretized fluxes and k-eigenvalue as outputs. Then, denote that a particular realization of the k-eigenvalue and scalar fluxes generated by a random perturbation of cross-section values by $k(\xi^i)$ and $\phi_g(\xi^i)$for $i = 1, \cdots, m$. For simplicity, denote a collection of scalar fluxes as:

$$\phi(\xi^i) = \begin{bmatrix} \phi_1(\xi^i)^T & \phi_2(\xi^i)^T & \cdots & \phi_G(\xi^i)^T \end{bmatrix}^T. \tag{4.34}$$

Then, $\phi(\xi^i)$ can be expressed by POD and the chaos expansion:

$$\phi(\xi^i) \approx \sum_{j=0}^{M} \sum_{k=1}^{r} \alpha_{jk} \Psi_j(\xi^i) \Phi_k \tag{4.35}$$

where Φ_k is an orthonormal vector generated by the POD procedure. The purpose of the POD approach is that it can reduce the computational cost significantly. If POD is not applied, then the chaos expansion must be applied to each nodal point for each energy group, which results in the number of unknowns begin $(M+1)nG$ whereas the number

of unknowns with POD is $(M+1)r$. This reduction represents a significant saving in cost and time. Using the orthogonality of Φ_k, Eq.(4.35) can be written as:

$$\langle \Phi_k, \phi(\xi^i) \rangle = \sum_{j=0}^{M} \alpha_{jk} \Psi_j(\xi^i) \text{ for } k = 1, \cdots r. \tag{4.36}$$

Now, Eq.(4.36) can be summarized as a matrix equation of the form $\mathbf{VA} = \mathbf{B}$ where

$$\mathbf{V} = \begin{bmatrix} \Psi_0(\xi^1) & \Psi_1(\xi^2) & \cdots & \Psi_M(\xi^1) \\ \Psi_0(\xi^2) & \Psi_1(\xi^2) & \cdots & \Psi_M(\xi^2) \\ \vdots & \cdots & \ddots & \vdots \\ \Psi_0(\xi^m) & \Psi_1(\xi^m) & \cdots & \Psi_M(\xi^m) \end{bmatrix} \tag{4.37}$$

$$\mathbf{A} = \begin{bmatrix} \alpha_{01} & \alpha_{02} & \cdots & \alpha_{0r} \\ \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1r} \\ \vdots & \cdots & \ddots & \vdots \\ \alpha_{M1} & \alpha_{M2} & \cdots & \alpha_{Mr} \end{bmatrix} \tag{4.38}$$

and

$$\mathbf{B} = \begin{bmatrix} \langle \Phi_1, \phi(\xi^1) \rangle & \langle \Phi_2, \phi(\xi^1) \rangle & \cdots & \langle \Phi_r, \phi(\xi^1) \rangle \\ \langle \Phi_1, \phi(\xi^2) \rangle & \langle \Phi_2, \phi(\xi^2) \rangle & \cdots & \langle \Phi_r, \phi(\xi^2) \rangle \\ \vdots & \cdots & \ddots & \vdots \\ \langle \Phi_1, \phi(\xi^m) \rangle & \langle \Phi_2, \phi(\xi^m) \rangle & \cdots & \langle \Phi_r, \phi(\xi^m) \rangle \end{bmatrix}. \tag{4.39}$$

In addition, the k-eigenvalue can be expanded using a chaos expansion:

$$k(\xi_i) = \sum_{j=0}^{M} k_j \Psi_j(\xi^i). \tag{4.40}$$

and Eq.(4.40) can be expressed as a matrix equation of the form $\mathbf{V}a = b$ where

$$a = \begin{bmatrix} k_0 & k_1 & \cdots & k_M \end{bmatrix}^T \tag{4.41}$$

and

$$b = \begin{bmatrix} k(\xi^1) & k(\xi^2) & \cdots & k(\xi^m) \end{bmatrix}. \tag{4.42}$$

Since the both matrix equations have the same matrix $\mathbf{V}$, they can be solved at the same

time,

$$\mathbf{V}\begin{bmatrix}\mathbf{A} & a\end{bmatrix} = \begin{bmatrix}\mathbf{B} & b\end{bmatrix}. \tag{4.43}$$

It has been recommended m to be $2M$ to solve Eq.(4.43) accurately [32]. This oversampling is mathematically justified by considering the relation between the Monte Carlo method and the Vandermonde-like matrix of multivariate orthogonal polynomials discussed in the following section.

4.2.1 A Property of Multivariate Orthonormal Vandermonde-like Matrices

A matrix $\mathbf{V}$ is commonly called a Vandermonde-like matrix. A univariate Vandermonde-like matrix is defined as:

$$\mathbf{V}_u = \begin{bmatrix} p_0(x_1) & p_1(x_1) & \cdots & p_m(x_1) \\ p_0(x_2) & p_1(x_2) & \cdots & p_m(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ p_0(x_n) & p_1(x_n) & \cdots & p_m(x_n) \end{bmatrix} \tag{4.44}$$

where $p_i(x_j)$ is i-th order polynomial of x_j. The properties of a Vandermonde-like matrix of three term recursion are well-known and there exists an efficient algorithm to solve a matrix equation with a Vandermonde matrix. However, a multivariate Vandermonde-like matrix is not well-studied to the best knowledge of the author. To examine the basic properties of a multivariate Vandermonde-like matrix, define an orthonormal multivariate Vandermonde-like matrix as:

$$\mathbf{V} = \begin{bmatrix} \Psi_0(\xi^1) & \Psi_1(\xi^1) & \cdots & \Psi_M(\xi^1) \\ \Psi_0(\xi^2) & \Psi_1(\xi^2) & \cdots & \Psi_M(\xi^2) \\ \vdots & \cdots & \ddots & \vdots \\ \Psi_0(\xi^m) & \Psi_1(\xi^m) & \cdots & \Psi_M(\xi^m) \end{bmatrix} \tag{4.45}$$

where $\Psi_i(\xi)$ is an orthonormal polynomial such that

$$\int_{\mathcal{Y}} \mathrm{d}\xi \; w(\xi)\Psi_i(\xi)\Psi_j(\xi) = \delta_{ij} \tag{4.46}$$

Then, the following theorem holds.

Theorem 2 (Diagonal Scaling for an orthonormal Vandermonde-like matrix)**.** *There exists* $\mathbf{D} \in \mathbb{R}^{m\times m}$ *for a multivariate orthonormal Vandermonde-like matrix such that*

$$\lim_{m\to\infty} \kappa_2(\mathbf{DV}) = \lim_{m\to\infty} ||\mathbf{DV}||_2 \left|\left|(\mathbf{DV})^\dagger\right|\right|_2 = 1 \tag{4.47}$$

where $\kappa_2(\mathbf{DV})$ *is a 2-norm condition number for* $\mathbf{DV}$*.*

Proof. Assume that a stochastic random vector ξ^i is sampled from the probability distribution $p(\xi) \neq 0$ where $\xi \in \mathcal{Y}$. Now, Eq.(4.46) can be rewritten using the Monte Carlo method with importance sampling,

$$\begin{aligned}
\int_{\mathcal{Y}} \mathrm{d}\xi\ w(\xi)\Psi_i(\xi)\Psi_j(\xi) &= \int_{\mathcal{Y}} \mathrm{d}\xi\ \frac{p(\xi)}{p(\xi)} w(\xi)\Psi_i(\xi)\Psi_j(\xi) \\
&= \lim_{m\to\infty} \frac{V}{m} \sum_{k=1}^{m} \frac{w(\xi^k)}{p(\xi^k)} \Psi_i(\xi_k)\Psi_j(\xi^k) \\
&= \frac{V}{m} \sum_{k=1}^{m} \frac{w(\xi^k)}{p(\xi^k)} \Psi_i(\xi^k)\Psi_j(\xi^k) + \mathcal{O}\left(\frac{1}{\sqrt{m}}\right)
\end{aligned} \tag{4.48}$$

Now, assume that the diagonal components of $\mathbf{D}$ are given by $[\mathbf{D}]_{ii} = \sqrt{\frac{Vw(\xi^i)}{mp(\xi^i)}}$. Then, the theorem can be proven by showing that the singular values of $\lim_{m\to\infty} \mathbf{DV}$ is 1. Assume a reduced singular value decomposition of $\mathbf{DV} = \mathbf{U\Sigma Q}^*$. Then,

$$\mathbf{V}^*\mathbf{D}^*\mathbf{DV} = \mathbf{V}^*\mathbf{D}^2\mathbf{V} = \mathbf{Q\Sigma}^2\mathbf{Q}^* \tag{4.49}$$

Now, (i,j)-th component of $\mathbf{V}^*\mathbf{D}^2\mathbf{V}$ is given by

$$\begin{aligned}
(\mathbf{V}^*\mathbf{D}^2\mathbf{V})_{ij} &= \frac{V}{m} \sum_{k=1}^{m} \frac{w(\xi^k)}{p(\xi^k)} \Psi_i(\xi^k)\Psi_j(\xi^k) \\
&= \mathcal{O}\left(\frac{1}{\sqrt{m}}\right) + \int_{\mathcal{Y}} \mathrm{d}\xi\ w(\xi)\Psi_i(\xi)\Psi_j(\xi) \\
&= \delta_{ij} + \mathcal{O}\left(\frac{1}{\sqrt{m}}\right)
\end{aligned} \tag{4.50}$$

Therefore,

$$\lim_{m\to\infty} (\mathbf{V}^*\mathbf{D}^2\mathbf{V})_{ij} = \delta_{ij} \Rightarrow \lim_{m\to\infty} \mathbf{V}^*\mathbf{D}^2\mathbf{V} = \mathbf{I} \tag{4.51}$$

which implies that $\lim_{m\to\infty} \mathbf{\Sigma} = \mathbf{I}$. Hence,

$$\lim_{m\to\infty} \kappa_2(\mathbf{DV}) = 1 \tag{4.52}$$

□

Since diagonal scaling does not change the solution of the linear system, it is natural to consider the diagonally scaled Vandermonde-like system based on Theorem 2:

$$\mathbf{DV}\begin{bmatrix}\mathbf{A} & a\end{bmatrix} = \mathbf{D}\begin{bmatrix}\mathbf{B} & b\end{bmatrix} \tag{4.53}$$

Theorem 2 justifies the oversampling in [32]. If a sufficient number of samples is collected, then Theorem 2 guarantees that it converges to the exact gPC coefficients. In addition, consider the normal equations of the Vandermonde-like system:

$$\mathbf{V}^*\mathbf{D}^2\mathbf{V}\begin{bmatrix}\mathbf{A} & a\end{bmatrix} = \mathbf{V}^*\mathbf{D}^2\begin{bmatrix}\mathbf{B} & b\end{bmatrix} \tag{4.54}$$

Since the off-diagonal components of $\mathbf{V}^*\mathbf{D}^2\mathbf{V}$ in Eq.(4.54) approach zero if m is sufficiently large, one may neglect them. Then, such an approximation becomes equivalent to the Monte Carlo method and if the samples are collected by advanced methods such as Latin Hypercube, it is equivalent to the Quisi-Monte Carlo method. However, neglecting the off-diagonal components makes the solution to Eq.(4.54) erroneous. Therefore, the Vandermonde-like system is more accurate than the Monte Carlo method in general.

Sample Node Distribution

The Askey scheme gives information on an optimal orthogonal basis function based on the probability density distribution of a stochastic variable [31]. However, the Askey scheme does not give any information on how the stochastic variables are sampled for non-intrusive polynomial chaos. A similar research has been conducted in approximation theory over centuries and sufficient knowledge exists to address these problems.

For Legendre polynomials, non-intrusive gPC can be considered as a multivariate interpolation problem since it interpolates responses by a specific class of polynomials. The accuracy of an interpolation is dramatically influenced by a node set. In approximation theory, how well an interpolation approximates a function with a give node set

is given by the Lebesgue constant $\Lambda_n(X)$ where X denotes the node set [2]. A Lebesgue constant measures how accurate a polynomial interpolation is with a given node set and polynomials, compared with the optimal interpolant of the same (or lower) degree in the maximum norm:

$$\left|\left|f - P_N^{\text{interp}}(X)\right|\right|_\infty \leq (1 + \Lambda_N(X)) \left|\left|f - P_N^{\text{optimal}}\right|\right|_\infty \tag{4.55}$$

where X denotes the node distribution, $P_N^{\text{interp}}(X)$ denotes the polynomial interpolation of order N, and P_N^{optimal} is the optimal interpolant of an order of N (or less). In order for a polynomial interpolant to be accurate, $\Lambda_n(X)$ should be as small as possible. It is important to note that Theorem 2 allows us to modify the sampling probability distribution of ξ_i as desired because Theorem 2 guarantees the convergence of non-intrusive gPC expansion regardless of what kind of sampling probability distribution is used.

The value of $\Lambda_n(X)$ depends in a very sensitive way on the node distribution and the type of a polynomial basis function. For example, in non-intrusive gPC expansion, the Legendre polynomials are used for a uniform distribution. Therefore, assume that we choose the equidistat node distribution as the sampling distribution. Now, the Lebesgue constant for the Legendre polynomial is given by [2]:

$$\Lambda_n(X_{\text{eq}}) = \mathcal{O}\left(\frac{2^n}{n \ln n}\right) \tag{4.56}$$

which grows exponentially. This Lebesgue constant shows that an interpolation by the chaos expansion with equidistat node distribution could be totally inaccurate and the accuracy decreases as the number of samples increases. Therefore, even if a stochastic variable in the original problem is uniformly distributed, it does not imply a stochastic variable should also be sampled uniformly. Rather, a stochastic variable should be sampled so that it minimizes a Lebesgue constant.

An optimal node distribution for a given polynomial is still under investigation. However, for Legendre and Chebyshev polynomials, sufficient information is available for our applications. The smallest possible Lebesgue constant is shown as [2]

$$\Lambda_n(X_{\min}) = \frac{2}{\pi}\left(\ln n + \gamma + \ln \frac{4}{\pi}\right) + \mathcal{O}(1). : \tag{4.57}$$

where γ is given by

$$\Gamma(1-x) = 1 + \gamma x + \mathcal{O}(x^2). \tag{4.58}$$

We want a node distribution that is as close as possible to Eq.(4.57). The Chebyshev node distribution has been considered as a good candidate. For the Legendre polynomials with the Chebyshev node distribution is $\mathcal{O}(\sqrt{n})$ which is better than Eq.(4.56). Moreover, the Chebyshev polynomials with the Chebyshev node distribution is known tobe semi-optimal [2]:

$$\Lambda_n(X_{\min}) = \frac{2}{\pi}\left(\ln n + \gamma + \ln\frac{8}{\pi}\right) + \mathcal{O}(1). \tag{4.59}$$

which is almost near the optimal value. From this fact, it is natural to consider that instead of using the Legendre polynomials, the Chebyshev polynomials with the Chebyshev node distribution will be more accurate than the Legendre polynomials, and the Chebyshev polynomial can be projected back to the Legendre polynomial easily.

Unfortunately, the above argument does not consider the multivariate cases. When the number of variables increases, the sampling approaches suffer from the curse of dimensionality. Assume that the number of stochastic variables is s. Then, if n Chebyshev nodal points are sampled in each stochastic variable, then the total number of samples is n^s (assuming full-tensor expansion). As s increases, the number of samples is unacceptably large. Therefore, the Chebyshev node distribution cannot be directly used in multivariate cases unless the cost of sampling is cheap.

An intuitive aid to the curse of dimensionality is to randomly sample stochastic variables according to a probability density function that simulates the Chebyshev node distribution. The Chebyshev node distribution is discrete, but there exists a continuous probability density function corresponding to the Chebyshev node distribution. The Chebyshev node distribution is given by:

$$x_j = -\cos\left(\frac{\pi j}{N}\right) \quad \text{for } i = 0, 1, \cdots, N \tag{4.60}$$

and the corresponding continuous probability density function is given by:

$$\mu(x) = \frac{1}{\pi\sqrt{1-x^2}}. \tag{4.61}$$

Now, one can sample x according to Eq.(4.61) randomly instead of using Eq.(4.60). Then,

Eq.(4.61) can be easily extended to multivariate cases:

$$\mu(x_1, x_2, \cdots, x_s) = \prod_{i=1}^{s} \frac{1}{\pi\sqrt{1-x_i^2}}. \tag{4.62}$$

Now, Eq.(4.62) can be used to sample nodal points randomly. If a sufficient number of samples is collected, then it will be close to the Chebyshev node distribution.

4.3 Sparse Approximation

Now, the nodal points can be sampled randomly using the Chebyshev continuous probability density distribution. However, non-intrusive gPC still requires many evaluations of the existing code, which could be computationally expensive. In nuclear engineering applications, each simulation code takes from hours to days.

How can gPC be used with the limited number of samples accurately? This question can be answered when the coefficients of the gPC expansion is *sparse*, where only a few terms of them are dominant [33]. Assume that only a few terms of the coefficients are important and the rest of the terms are negligible or close to 0. Then, the orthonormal Vandermonde-like system can be treated as an optimization problem described as a P_0 minimization problem given by:

$$(P_0): \qquad \min ||x||_0 \quad \text{subject to} \quad \mathbf{V}x = b \tag{4.63}$$

where $||x||_0$ is the number of non-zero entries, defined as 0-norm, and $\mathbf{V}$ is an orthonormal Vandermonde-like matrix. The properties of sparsity approximation can be found in a textbook, eg. [33]. The application of the sparse approximation in the non-instructive gPC can be found in [34].

Orthogonal Marching Pursuit

Our primary interest is in how this P_0 optimization problem can be solved. One famous algorithm to solve the P_0 problem is the *Orthogonal Marching Pursuit* (OMP) [33]. OMP is considered to be a greedy algorithm because it tries to find a local optimizer to minimize the residual and to update a solution until the residual meets the user-specified tolerance. In OMP, the method tries to find the support of a solution $\mathcal{S}$ by sweeping all components

of a solution individually and to quantify the value over the support by the least-squares problem. OMP can be summarized as follows:

First, consider that the initial guess to Eq.(4.63) is $x_0 = 0$ and the initial support $\mathcal{S}_0 = \varnothing$ is an empty set. Now, the residual to the initial guess is given by $r_0 = b - \mathbf{V}x_0 = b$. Then, we are interested in a scalar multiplication of a canonical vector that minimizes the initial residual r_0. So, find the support of the solution to minimize the residual by solving:

$$\text{Find } j \text{ such that } \min_j ||\mathbf{V}\hat{x}_j - r_0|| \tag{4.64}$$

where $\hat{x}_j$ is a scalar multiplication of a canonical vector $\hat{e}_j$. This can be done greedily by sweeping all j and finding the minimal value of $||\mathbf{V}\hat{x}_j - r_0||$. Denote the j that minimizes $||\mathbf{V}\hat{x}_j - r_0||$ as j_1 and add it to the support:

$$\mathcal{S}_1 = \mathcal{S}_0 \cup \{j_1\} \tag{4.65}$$

Then, obtain a sparse solution for the original system on the new support $\mathcal{S}_1$ by solving the least-squares problem:

$$\min_{x_{\mathcal{S}_1}} ||\mathbf{V}_{\mathcal{S}_1} x_{\mathcal{S}_1} - b||_2 \tag{4.66}$$

where $\mathbf{V}_{\mathcal{S}_1}$ is defined as a matrix of size $m \times |\mathcal{S}_1|$ that contains the columns from $\mathbf{V}$ that belong to the support $\mathcal{S}_1$ and $x_{\mathcal{S}_1}$ is defined as a vector of size $|\mathcal{S}_1| \times 1$. Then, $x_{\mathcal{S}_1}$ is projected onto the support $\mathcal{S}_1$ of x denoted by x_1. Now, the residual is updated with new candidate solution, $r_1 = b - \mathbf{V}x_1$. Then, the same procedure is repeated until the user-specified tolerance is met. The procedure for OMP is summarized in Algorithm 5.

Algorithm 5 Orthogonal Marching Pursuit

Given a $m \times P$ multivariate Vandermonde-like matrix $\mathbf{V}$ and $P \times 1$ vector b, a tolerance ϵ, the following algorithm computes a sparse approximation solution for $\mathbf{V}x = b$.

1. Set $k = 0$.
2. $x_0 = 0$.
3. $r_0 = b - \mathbf{V}x_0$.
4. $\mathcal{S}_0 = \varnothing$.
5. **while** $||r_k||_2 \geq \epsilon$, repeat the following steps:
   (a) $k = k + 1$.
   (b) Find j such that $\min_j ||\mathbf{V}\hat{x}_j - r_{k-1}||$
   (c) Update the support $\mathcal{S}_k = \mathcal{S}_{k-1} \cup \{j\}$
   (d) Find the new solution by $\min_{x_{\mathcal{S}_k}} ||\mathbf{V}_{\mathcal{S}_k} x_{\mathcal{S}_k} - b||_2$
   (e) Update the solution x_k by $x_{\mathcal{S}_k}$.
   (f) Calculate the new residual $r_k = b - \mathbf{V}x_k$
6. Output: x_k.

As can be seen in Algorithm 5, OMP is a greedy algorithm and not efficient. When P is large, then OMP takes long time to compute.

Dantzig Selector

When the number of available samples is small, then OMP may not be viable. In this case, a common alternative to OMP is l_1 optimization approach (also commonly P_1

optimization) described as:

$$(P_1) \quad \min_x ||x||_1 \text{ Subject to } b = \mathbf{V}x \tag{4.67}$$

Different from the P_0 problem, the P_1 problem tries to minimize the l_1 constraint. The l_1 problem is typically solved by *linearizing* the P_1 problem. Consider that the solution to Eq.(4.67) can be decomposed into two parts, $x = u - v$ where $u, v \in \mathbb{R}_+^P$ are both non-negative vectors. Denote that $z = \begin{bmatrix} u^T & v^T \end{bmatrix}^T$ and $\mathbf{1}$ as a vector with all components 1. Then, $||x||_1$ can be written as:

$$||x||_1 = \mathbf{1}^T(u + v) = \mathbf{1}^T z. \tag{4.68}$$

Likewise, $\mathbf{V}x$ can be written as:

$$\mathbf{V}x = \mathbf{V}(u - v) = \begin{bmatrix} \mathbf{V} & -\mathbf{V} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \mathbf{V} & -\mathbf{V} \end{bmatrix} z. \tag{4.69}$$

Therefore, Eq.(4.67) can be written in linear programming form:

$$\min_z \ \mathbf{1}z \quad \text{Subject to } b = \begin{bmatrix} \mathbf{V} & -\mathbf{V} \end{bmatrix} z \ \text{ and } \ z \geq 0. \tag{4.70}$$

Eq.(4.70) is considered to have a classical linear programming structure, which can be easily solved by the optimization toolbox in MATLAB. Recently, an efficient variant of Eq.(4.67) referred to as the *Dantzig Selector*, was proposed by Candes and Tao [35] (See [36] how to solve l_1 constraint linear problem). In the Dantzig selector, the P_1 problem is treated as:

$$\min_x ||x||_1 \quad \text{Subject to } \left|\left|\mathbf{V}^T(b - \mathbf{V}x)\right|\right|_1 \leq \epsilon \tag{4.71}$$

where ϵ is user-defined tolerance. Then, the Dantzig selector can be cast into linear programming form that can be easily solved using MATLAB. It is given by:

$$\min_{u,v} \mathbf{1}^T u + \mathbf{1}^T v \text{ Subject to } \begin{cases} u \geq 0 \\ v \leq 0 \\ \epsilon\mathbf{1} \leq \mathbf{V}^T(b - \mathbf{V}x) \leq \epsilon\mathbf{1} \end{cases} \tag{4.72}$$

where u is an auxiliary variable with $u \geq |x|$. Eq.(4.72) also can be solved by the optimization toolbox in MATLAB. Further, there is a third party MATLAB toolbox that can solve the Dantzig selector using the dual-primal interior-point method [37].

The sparse approximation is potentially capable of solving gPC with a very limited number of samples if the solution is sparse and highly applicable to nuclear reactor simulation in such a case.

4.4 Input Parameter Reduction

The sparse approximation can reduce the number of samples efficiently if and only if the response function is sparse with respect to the input parameters. However, when the number of input parameters is larger and they are not sufficiently sparse, then an alternative approach is *input parameter reduction*, where the number of input parameters is reduced by the sensitivity of the response with respect to the input parameters.

Assume the k-eigenvalue expressed in terms of the stochastic variables, $k(\xi)$. Now, the sensitivity of the k-eigenvalue with respect to each stochastic variable is given as:

$$\frac{\partial k(\xi)}{\partial \xi_i} = \text{Sensitivity of } k(\xi) \text{ w.r.t. } \xi_i. \tag{4.73}$$

Then, the sensitivity represents the importance of each stochastic variable at ξ_i. For instance, consider a case where a function only depends on two stochastic variables, $f(\xi_1, \xi_2)$ and each sensitivity at $\xi_1 = 1$ and $\xi_2 = 2$ is given by $\frac{\partial f(\xi_1,\xi_2)}{\partial \xi_1} = 10$ and $\frac{\partial f(\xi_1,\xi_2)}{\partial \xi_2} = 10^{-6}$. Now, when $(\xi_1, \xi_2) \approx (1, 2)$, there is little contribution from the stochastic variable, ξ_2. In such a case, the second stochastic variable is negligible. However, even though the second stochastic variable can be neglected, this is only applicable when $(\xi_1, \xi_2) \approx (1, 2)$. Therefore, the sensitivity information must be collected over the domain of the stochastic variables to identify the importance of each stochastic variable. Likewise, when many stochastic variables exist, the change in response only depends on a few combinations of stochastic variables.

Assume that the upper script denotes i-th sample and the sensitivity vector of the k-eigenvalue at $\xi = \xi^i$ is represented by:

$$S(\xi^i) = \left[\left.\frac{\partial k(\xi)}{\partial \xi_1}\right|_{\xi^i} \quad \left.\frac{\partial k(\xi)}{\partial \xi_2}\right|_{\xi^i} \quad \cdots \quad \left.\frac{\partial k(\xi)}{\partial \xi_n}\right|_{\xi^i} \right]^T \tag{4.74}$$

which represents the importance of each variable at $\xi = \xi^i$. Now, the sensitivity matrix for the k-eigenvalue is written as:

$$\mathbf{S} = \begin{bmatrix} S(\xi^1) & S(\xi^2) & \cdots & S(\xi^m) \end{bmatrix} \tag{4.75}$$

where m is the number of samples collected to obtain the sensitivity matrix. Now, to identify the dominant direction of the sensitivity matrix, take SVD on $\mathbf{S}$:

$$\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{4.76}$$

where $\mathbf{U}$ contains the information on the dominant sensitivity vector in each row, Σ contains the magnitude or importance of each dominant sensitivity vector in $\mathbf{U}$, and each column of $\mathbf{\Sigma}\mathbf{V}^*$ contains information on how $\mathbf{U}$ can represents $S(\xi^i)$. Note that $\mathbf{U}$ is called the **active subspace** of the k-eigenvalue for the input parameter space because any other orthogonal directions to $\mathbf{U}$ has little contribution in k-eigenvalue [38].

Once the important directions for the stochastic variables are identified, then the k-eigenvalue can be approximated by a small number of pseudo-stochastic variables given by:

$$\alpha = \mathbf{U}_r^* \xi \tag{4.77}$$

where $\mathbf{U}_r$ is the first r dominant columns of $\mathbf{U}$. Now, the k-eigenvalue can be approximated as:

$$k(\xi) \approx k(\mathbf{U}_r \alpha) = k(\alpha). \tag{4.78}$$

It is important to note that the error bound for the input parameter reduction for the k-eigenvalue is not known. Therefore, to validate the accuracy of the input parameter reduction, cross-validation must be performed. Otherwise, the input parameter reduction cannot be trusted. The common approach to validate the input parameter reduction is to benchmark $k(\mathbf{U}_r \alpha_i)$ against new set of $k(\xi_j)$ and one can get the statistical upper bound of the error using order statistics.

Another important remark is that when the original stochastic variables are reduced to the pseudo stochastic variables, the probability distribution of the original stochastic

variables is not preserved. That is, α will have different probability distribution. Therefore, gPC cannot be applied to the pseudo-stochastic variables. Instead, the k-eigenvalue is recovered with respect to each component of α by the polynomial interpolation described by:

$$k(\alpha) = \sum_{i=1}^{\infty} a_i \Psi_i(\alpha) \tag{4.79}$$

where a_i is the set of undetermined coefficients and $\Psi_i(\alpha)$ are multivariate orthogonal polynomials that can approximate $k(\xi)$. If the domain of α is bounded, then a Chebyshev polynomial is a good candidate. Then, an appropriate function must be chosen based on α. Now, the mean can be calculated by:

$$\mu(k(\xi)) = \frac{1}{V} \int d\xi k(\alpha(\xi)) = \frac{1}{V} \sum_{i=0}^{M} \int d\xi a_i \Psi(\alpha(\xi)) \tag{4.80}$$

and the variance can be calculated with the similar manner. As can be seen, the orthogonal properties of gPC are lost in the input parameter reduction. Therefore, analytic evaluation of the mean and variance is difficult and a numerical integration technique should be applied.

Chapter 5

Numerical Results for POD Methods

Numerical results are obtained by simulating the test models described in Chapter 2. For simplicity, the normalization constraint is changed to:

$$\phi^T \phi = 1. \tag{5.1}$$

Test Model A

To verify the proposed methods in the chapter 3, test model A, a 1D two-group simple fuel model is used. The cross-section values in Table 2.1 are assumed to have uncertainties. Each cross-section valueis assumed to be normally distributed with standard deviation of 10% of its reference value from the reference value. The thermal group absorption cross-section and the removal cross-section are assumed to be equal, and hence the total number of stochastic variables is 12. The model is spatially discretized with equal node width resulting in 1000 total nodes, and hence the dimension of $\mathbf{L}$ and $\mathbf{F}$ is 2000 (because the number of unknowns per node is equal to two, representing the two group flux). The generalized eigenvalue problem was solved by the power iterative method with a tolerance 10^{-12} for the residual.

First, the POD procedure is applied to this model. To apply the POD procedure, 100 random flux solutions, $\phi(\xi^i)$ are generated by randomly perturbing cross-section values . Fig. 5.1 and Fig. 5.2 show the plots of 100 realizations of fluxes. Both figures show each flux has a similar shape to different perturbations. Therefore, a solution should be

expressed by a linear combination of 100 flux solutions.

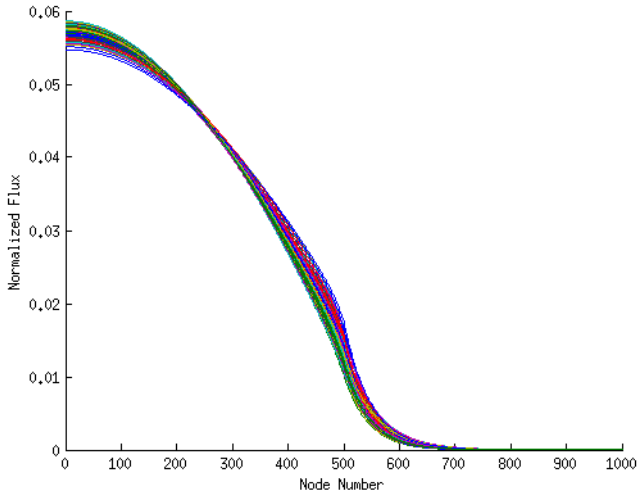

Figure 5.1: 100 Realizations of Fast Flux Solutions

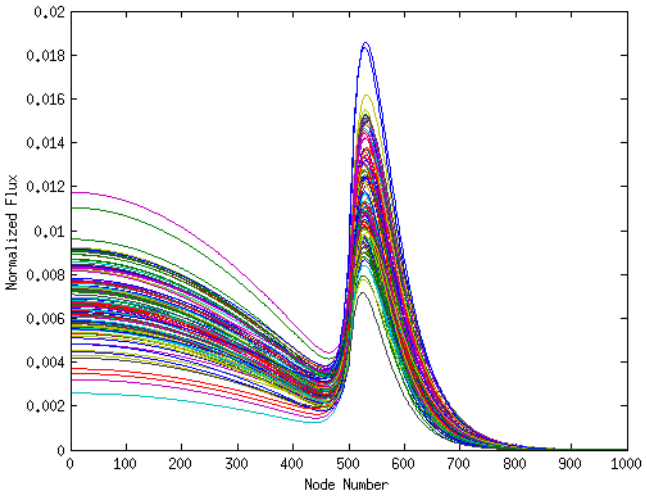

Figure 5.2: 100 Realizations of Thermal Flux Solutions

To verify this assumption, 100 extra random flux solutions, $\tilde{\phi}(\xi^j)$, are generated by randomly perturbing cross-section values. Fig. 5.3 shows the error given by the following equation,

$$\max_j \left|\left| \tilde{\phi}(\xi^j) - \sum_{i=1}^{r} a_{ij}\phi(\xi^j) \right|\right|_2 . \tag{5.2}$$

where r is the number of snapshots. It is obvious that when the number of snapshots increases, the L_2 error decreases exponentially. Even though the dimension of the original problem is 2000, the actual numerical rank of the problem is only about 50 based on Fig. 5.3.

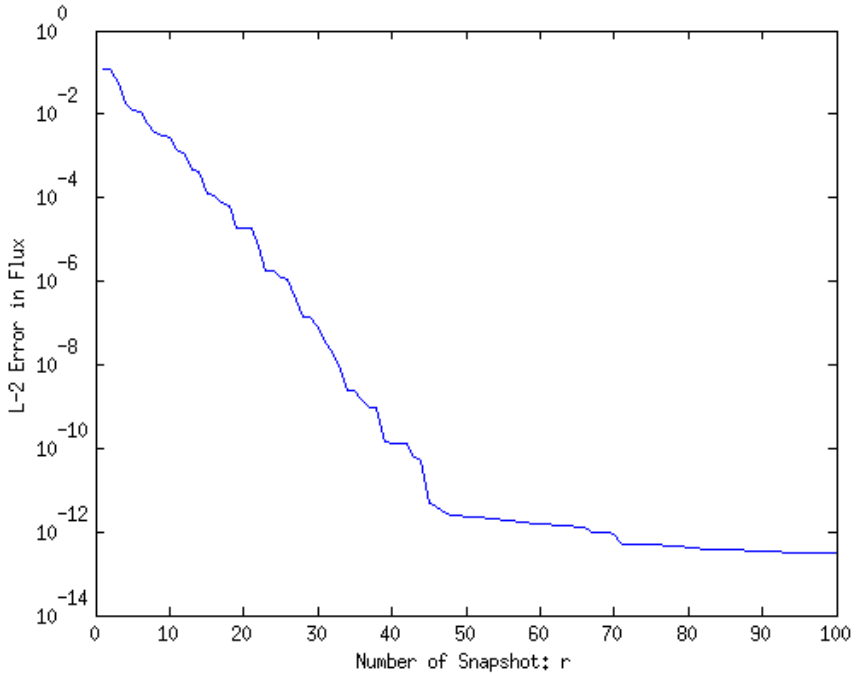

Figure 5.3: L_2 Error of the POD procedure with 100 samples

The same numerical test was performed on the adjoint solutions. 100 adjoint flux solutions $\phi^*(\xi^j)$ were generated from randomly perturbing cross-section values, and 100 extra adjoint flux solutions $\tilde{\phi}^*(\xi^j)$ were generated for the numerical test. Fig. 5.4 shows the error given by the following equation,

$$\max_j \left|\left| \tilde{\phi}^*(\xi^j) - \sum_{i=1}^{r} a_{ij}^* \phi^*(\xi^j) \right|\right|_2 . \tag{5.3}$$

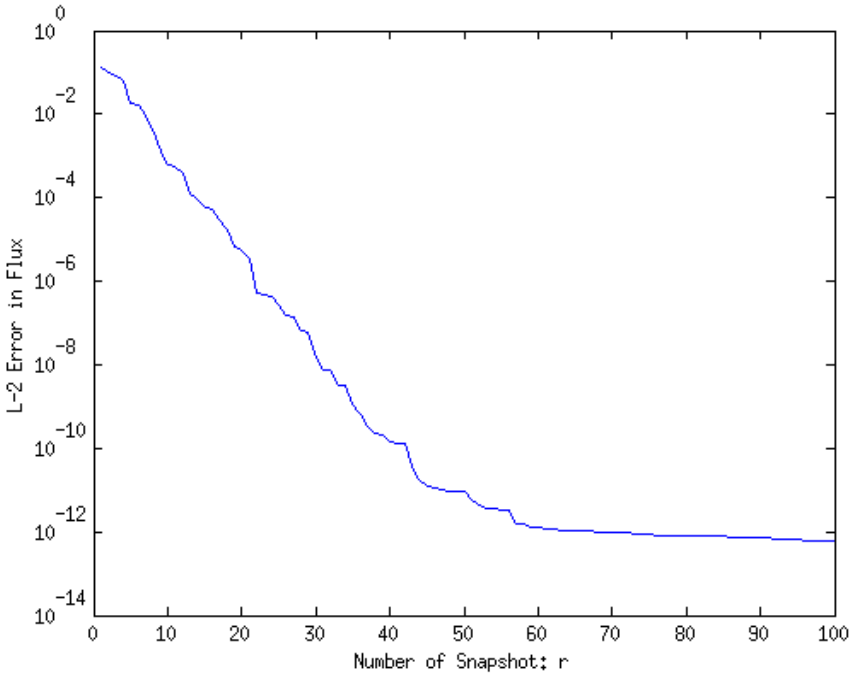

Figure 5.4: L_2 Error of the POD procedure with 100 samples for the Adjoint Flux

Fig. 5.4 shows almost the same result compared with the forward solutions. Therefore, the forward flux and the adjoint flux both live in the low-dimensional subspace. Hence, reduced order modeling is possible.

It is important to note that the numerical rank of the POD-basis does not change with the number of nodes. To demonstrate this property, the test model A was solved by different mesh sizes with the dimensions of $\mathbf{L}$ equal to 500, 1000, and 2000. Fig. 5.5 shows the POD-Error obtained by these mesh sizes. From the figure, the POD-Error does not depend on the mesh size. This fact allows us to determine the number of the POD-basis in a relatively coarse mesh and then use it for the fine mesh to minimize the number of executions. The same fact can be shown for the adjoint flux.

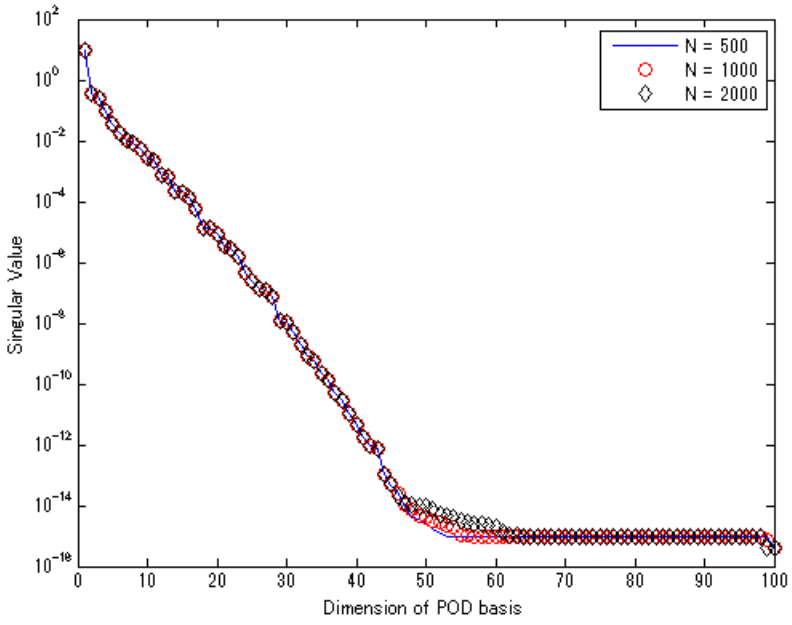

Figure 5.5: Effect of the Mesh Refinement on the POD-Error

Now, we need to find r and Φ_i to construct the reduced order model. As discussed in Chapter 3, the optimal POD-basis can be obtained from the singular value decomposition or the eigenvalue decomposition. Fig. 5.6 shows the POD-basis for the fast neutron flux. Fig. 5.7 shows the POD-basis for the thermal neutron flux. It can be seen that the POD-basis keeps the same level of the continuity that the solution of the problem. Since the scalar fluxes can be represented by the linear combination of the POD-bases, the POD-bases can also be represented by the linear combination of the scalar fluxes. Since the scalar fluxes have C^0 continuity, then the POD-basis should also have the C^0 continuity. This fact allows us to interpolate the coarse POD-basis to fine POD-basis using an appropriate interpolation scheme. In other words, the POD-basis can be used in the similar manner such as the Multi-grid approach. Also, the first three POD-bases for the adjoint solution are shown in Fig. 5.8 and Fig. 5.9

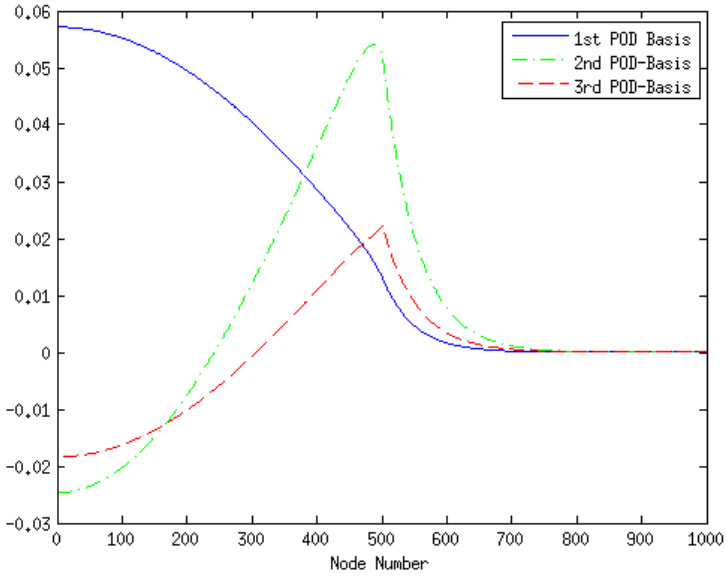

Figure 5.6: First Three POD-basis for the Forward Flux of the Fast Flux Part

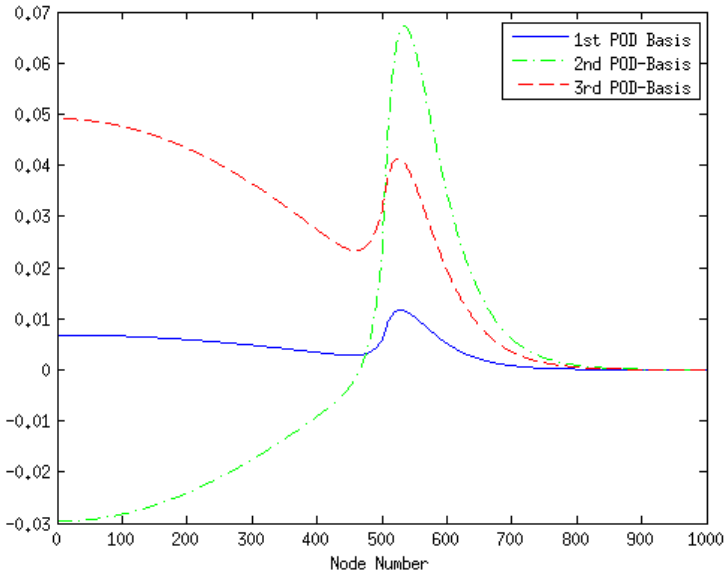

Figure 5.7: First Three POD-basis for the Forward Flux of the Thermal Flux Part

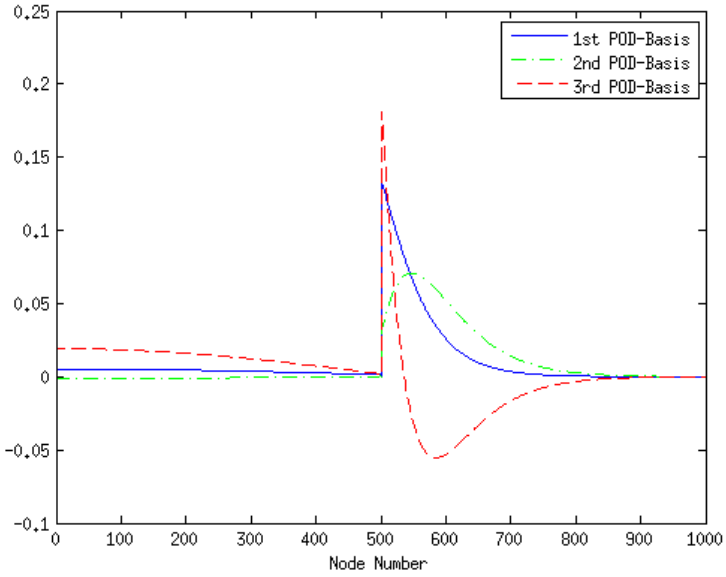

Figure 5.8: First Three POD-basis for the Adjoint Flux of the Fast Flux Part

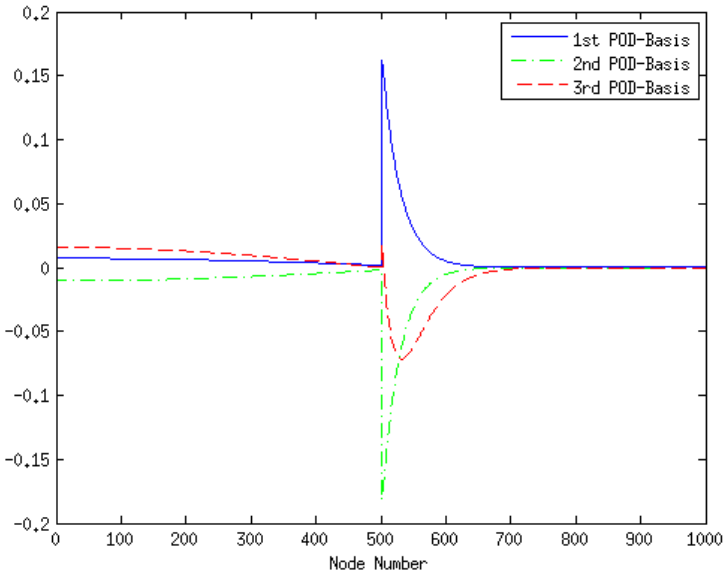

Figure 5.9: First Three POD-basis for the Adjoint Flux of the Thermal Flux Part

Using these POD-bases, the generalized eigenvalue problem can now be reduced using the POD-Galerkin method, the adjoint approach ($\gamma = 0$ and $\gamma = 0.0005$), and the k-LS method. Each method is benchmarked against 100 realizations of forward solutions generated by randomly perturbing cross-section values. That is,

$$\max_{i=1,\ldots,100} \left|\left|\phi(\xi^i) - \phi_{\text{POD}}(\xi^i)\right|\right|_2 . \tag{5.4}$$

Fig. 5.10 shows the worst relative L_2 norm errors in flux. As can be seen, as the dimension of the POD-basis increases, the L_2 norm errors exponentially decay. One of the interesting observations is that even though the POD-Galerkin approach is not guaranteed to work with the k-eigenvalue problem, in this problem it worked. Actually, it showed the best results for the flux errors. In addition, the adjoint approach ($\gamma = 0$) shows the worst performance in the flux calculation. The adjoint approach ($\gamma = 0.0005$) improved the result in the flux compared with the adjoint approach ($\gamma = 0$).

On the other hand, Fig. 5.10 shows the relative errors in the k-eigenvalue and the adjoint approach ($\gamma = 0$) shows the best results. The dimension of the generalized eigenvalue problem was reduced from 2000 to 16 to achieve errors on the order of machine precision. In other words, the k-eigenvalue has lower dimension of the active subspace than flux in terms of the number of state variables. In addition, the adjoint approach ($\gamma = 0.0005$) showed the second to the best performance. Note that the POD-Galerkin method showed the worst performance as expected.

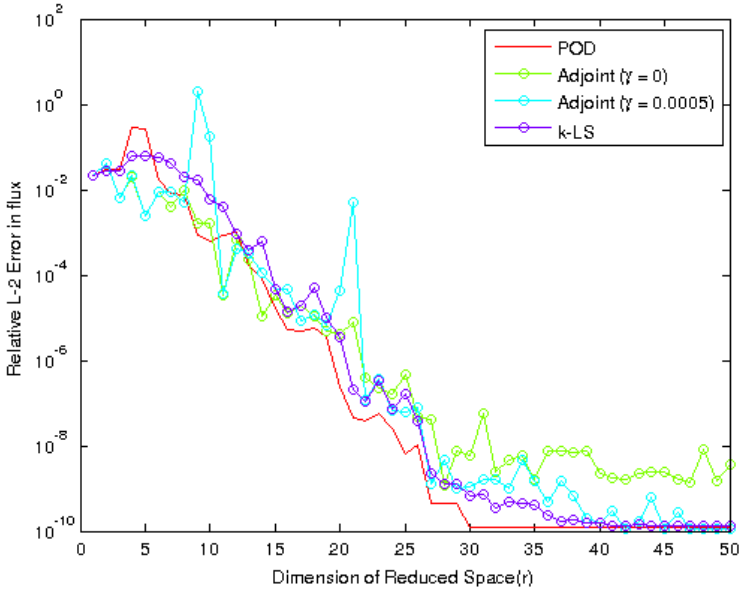

Figure 5.10: Worst Relative L_2 Norm Error in Flux with Different POD Approaches

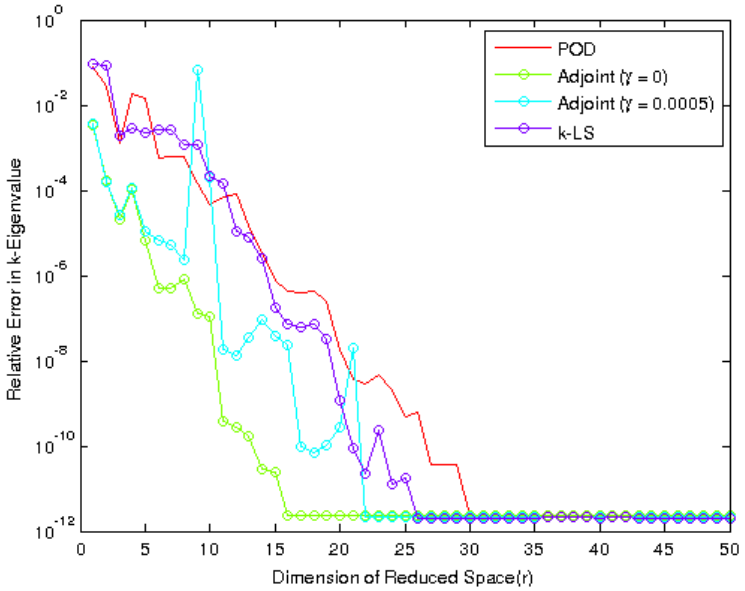

Figure 5.11: Worst Relative Error in k-Eigenvalue with Different POD Approaches

In addition, Fig. 5.12 and Fig. 5.13 show the performance of each method to calculate the flux at the dimension of the POD-basis equal to 15 and 30. When $r = 15$, the performance of each method is not so different. However, when $r = 30$, the POD-Galerkin method exhibit the best performance among all. In addition, the adjoint approach ($\gamma = 0.0005$) shows better results than the adjoint approach ($\gamma = 0$). Overall, the adjoint approach ($\gamma = 0$) is not well-suited to recovering the forward solution from the reduced space. On the other hand, Fig. 5.14 shows the performance of each method to calculate k-eigenvalue at the dimension of the POD-basis equal to 15. Opposite from the flux case, the adjoint approach ($\gamma = 0$) showed superior performance when finding the k-eigenvalue.

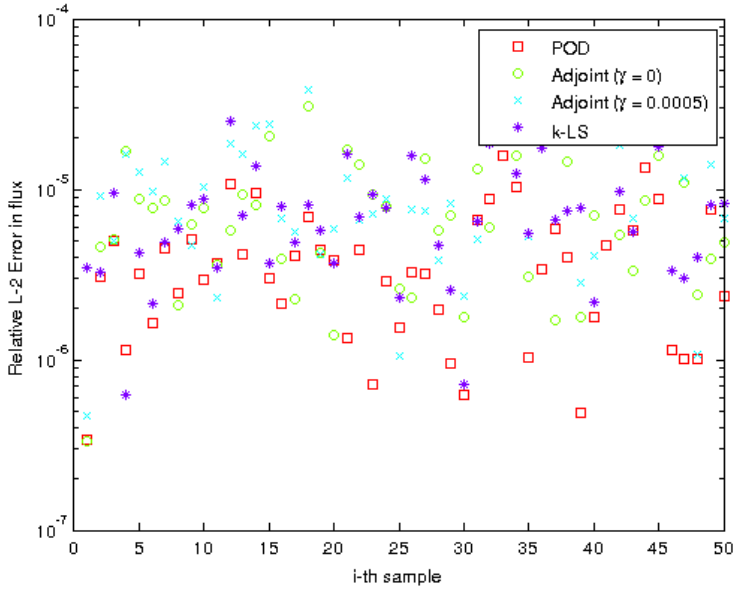

Figure 5.12: Relative L_2 Norm Errors in Flux with the POD-basis ($r = 15$)

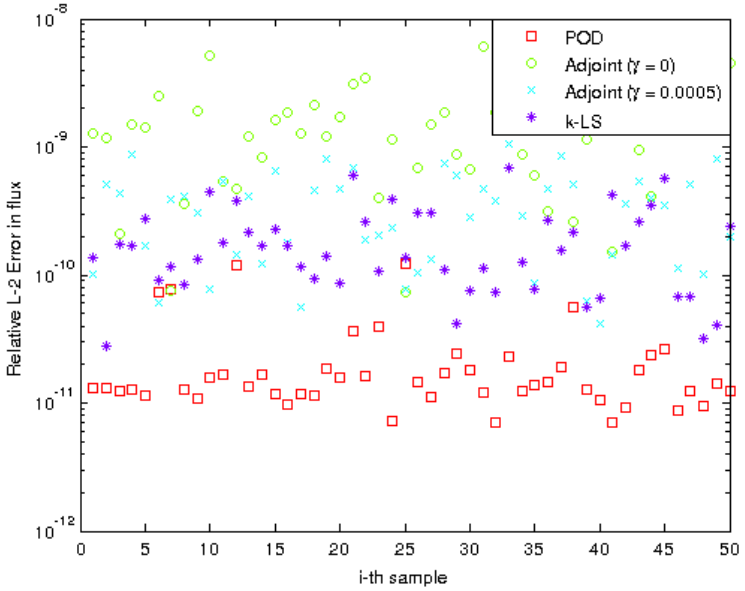

Figure 5.13: Relative L_2 Norm Errors in Flux with the POD-basis ($r = 30$)

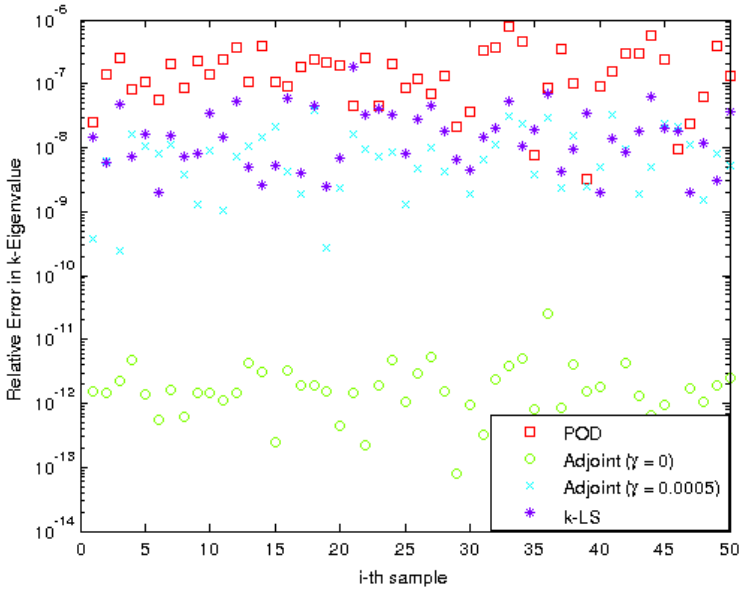

Figure 5.14: Relative Errors in k-Eigenvalue with the POD-basis ($r = 15$)

Test Model B

The same numerical test is also applied on test model B. In this model, the geometry is equally divided into 512 cells with equal width. The macroscopic cross-sections for each isotope are assumed to be normally distributed with standard derivation of 10% of its value. Fig. 5.15-5.18 show the realizations of 20 thermal and fast currents and fluxes. From the plots, it can be seen that currents and fluxes do not change their values with microscopic cross-section perturbations.

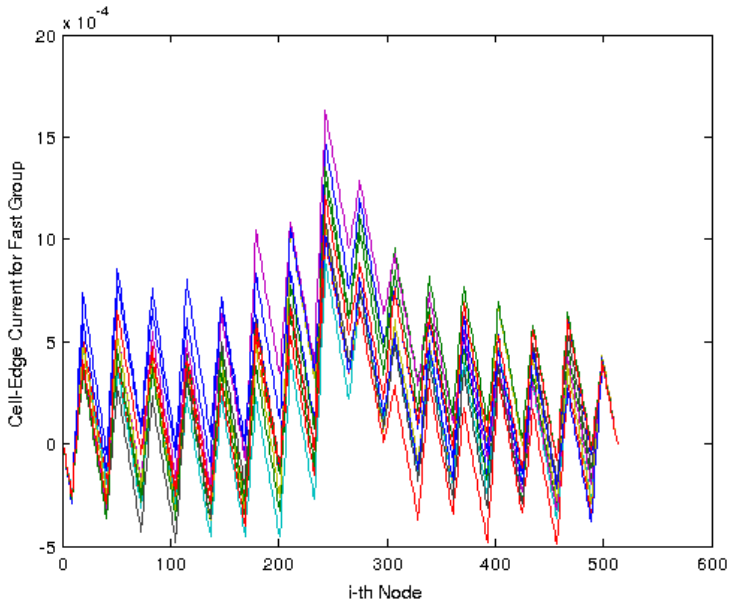

Figure 5.15: 20 Realizations of Cell-Edge Current in Fast Energy Group

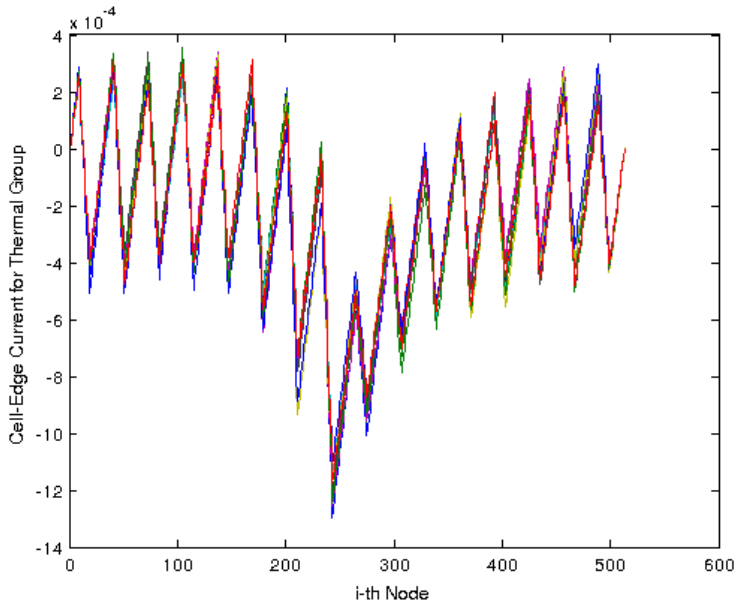

Figure 5.16: 20 Realizations of Cell-Edge Current in Thermal Energy Group

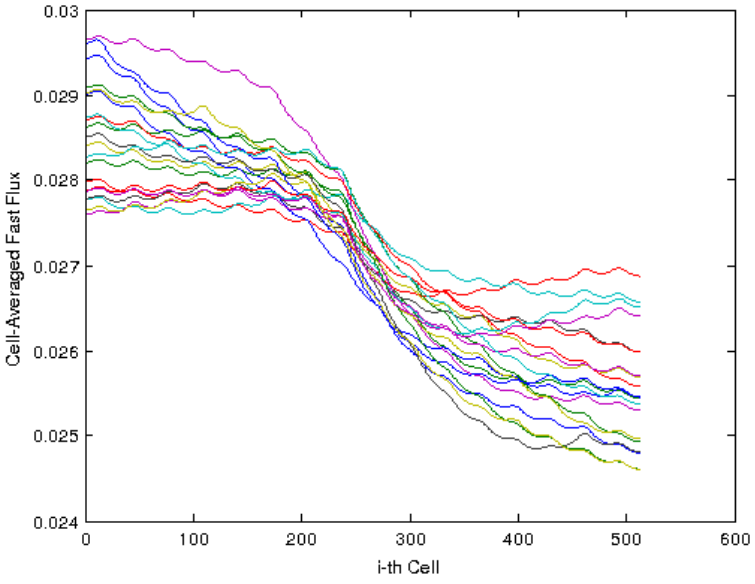

Figure 5.17: 20 Realizations of Cell-Averaged Fast Flux

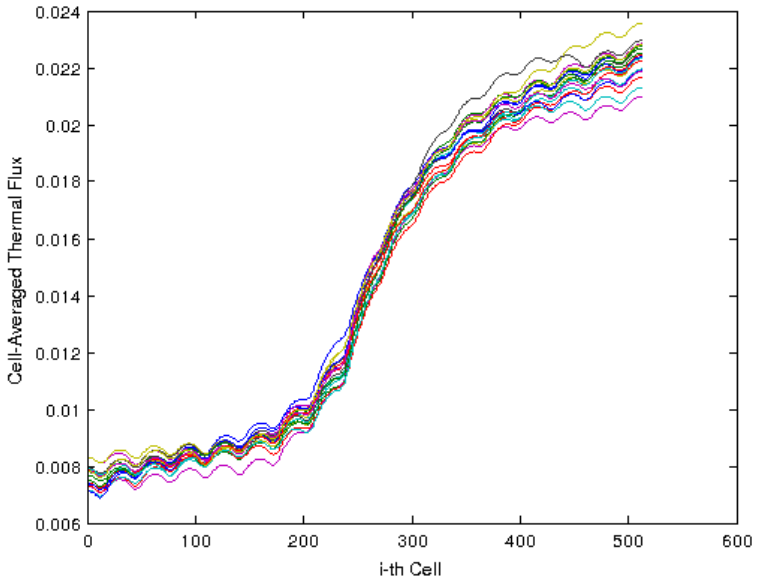

Figure 5.18: 20 Realizations of Cell-Averaged Thermal Flux

Fig. 5.19-5.22 show the adjoint solutions for the currents and fluxes. Recall that the adjoint solutions are obtained by simply transposing $\mathbf{L}$ and $\mathbf{F}$ operators. One important observation is that the adjoint solutions are close to zero almost everywhere, except at a few points (See the magnitude of Fig. 5.19-5.22). Even though these adjoint solutions are correct, when the generalized eigenvalue problem is projected onto these adjoint solutions, information on the fluxes and currents will be lost. This causes a serious problem when the eigenvector is recovered from the reduced space since such information is already lost. This can be partially remedied by using $\gamma > 0$.

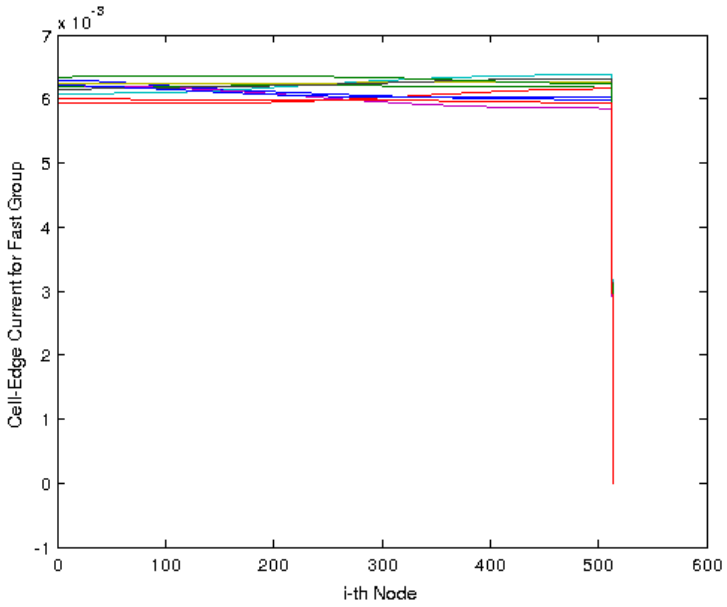

Figure 5.19: 20 Realizations of Adjoint Part of Cell-Edge Current in Fast Energy Group

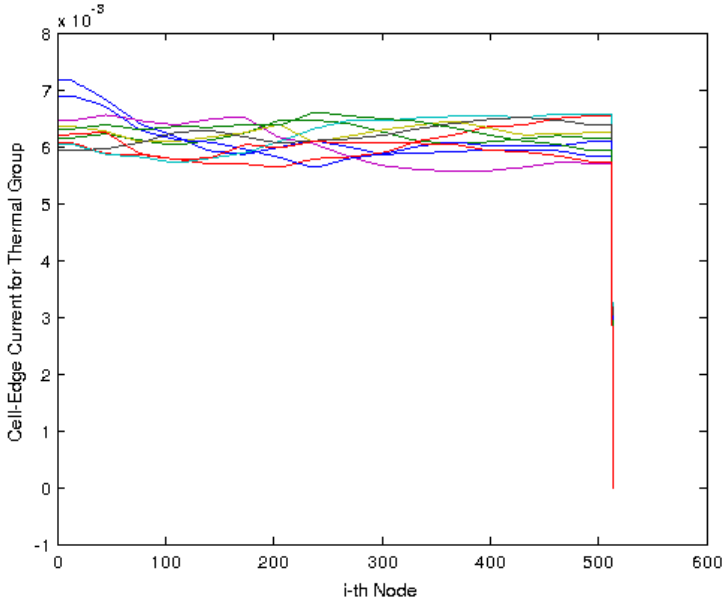

Figure 5.20: 20 Realizations of Adjoint Part of Cell-Edge Current in Thermal Energy Group

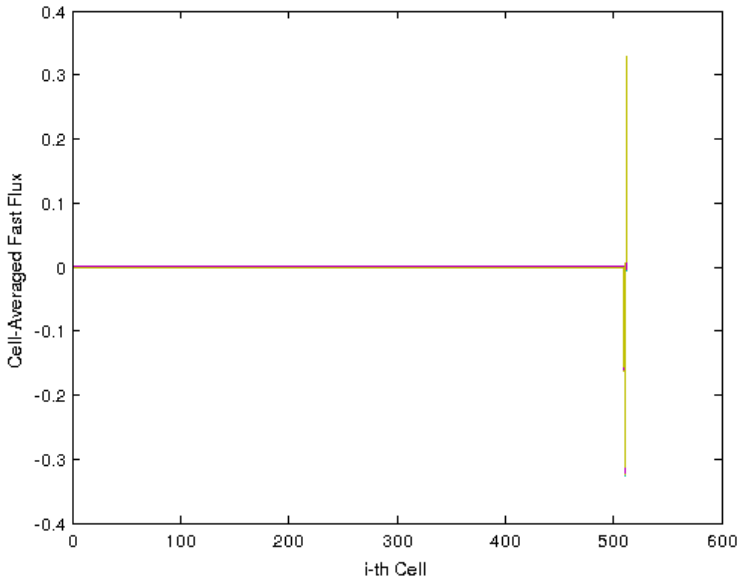

Figure 5.21: 20 Realizations of Adjoint Part of Cell-Averaged Fast Flux

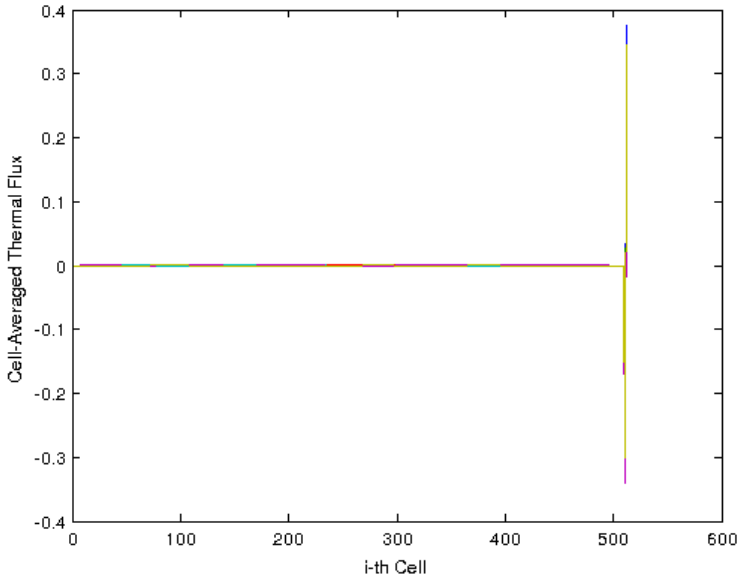

Figure 5.22: 20 Realizations of Adjoint Part of Cell-Averaged Thermal Flux

Now, Fig. 5.23-5.26 show the first three POD-bases of the forward solutions.

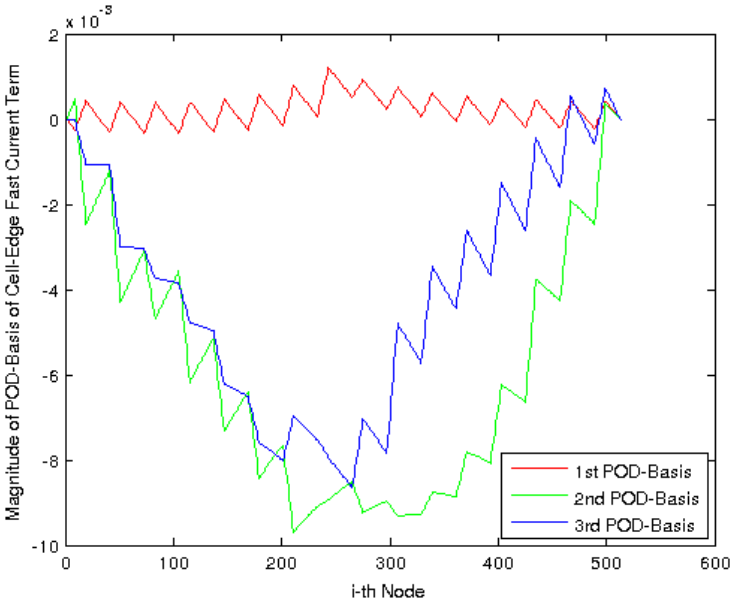

Figure 5.23: First Three POD-bases of Fast Current Part

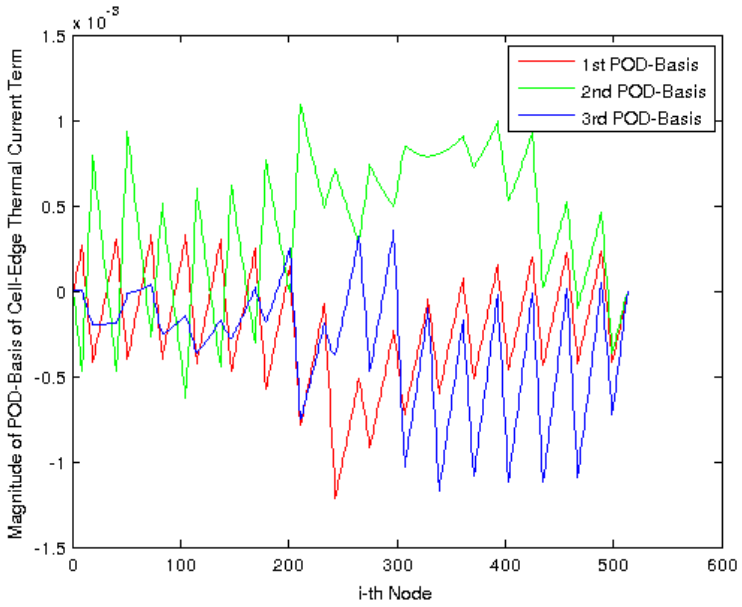

Figure 5.24: First Three POD-bases of Thermal Current Part

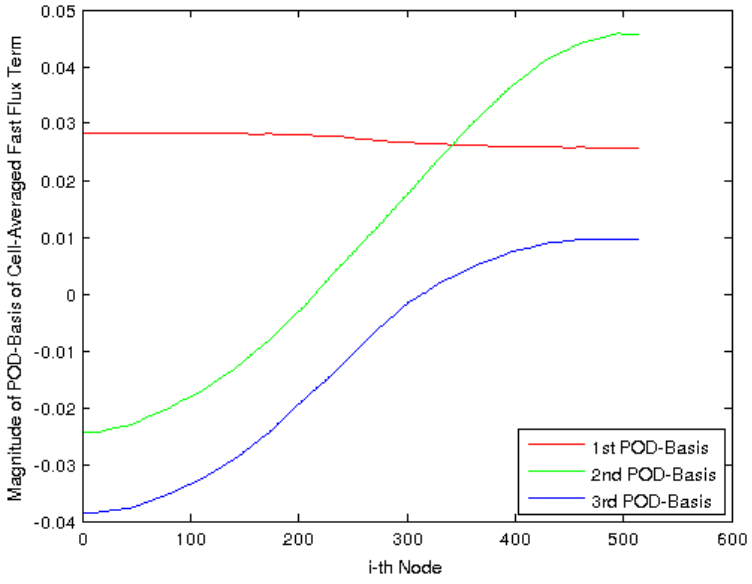

Figure 5.25: First Three POD-bases of Fast Flux Part

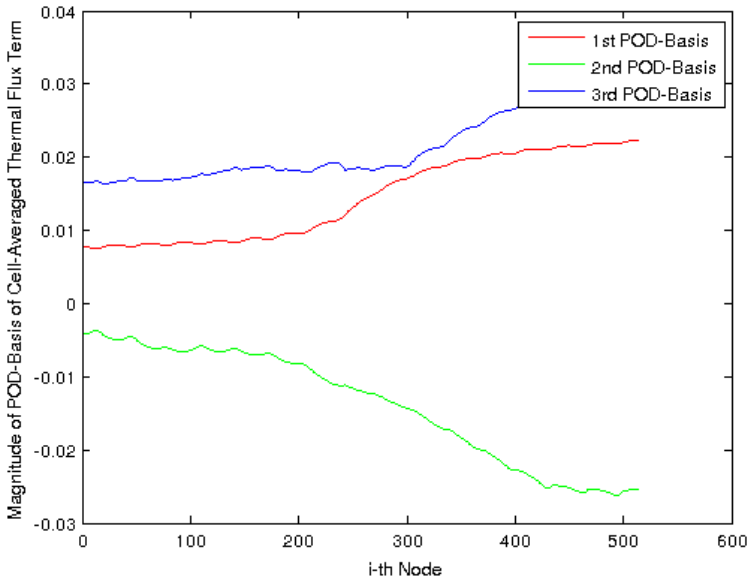

Figure 5.26: First Three POD-bases of Thermal Flux Part

Fig. 5.27-5.30 show the first three POD-bases for the adjoint solution. As can be expected from the adjoint solution, the majority of the POD-basis is almost zero and an only few of components are dominant.

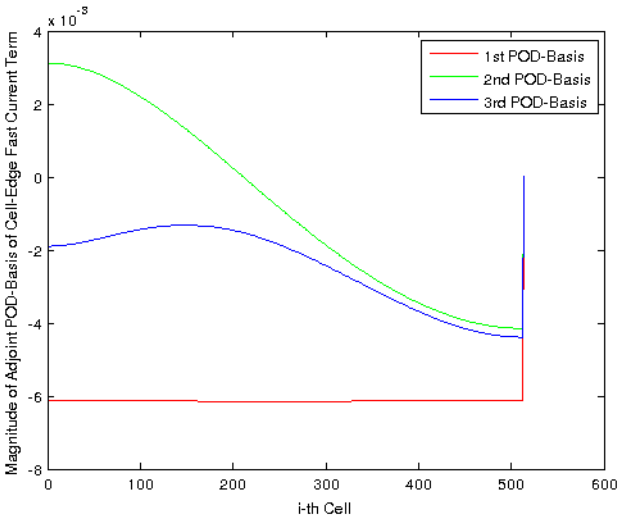

Figure 5.27: First Three POD-bases of Adjoint Solution of Fast Current

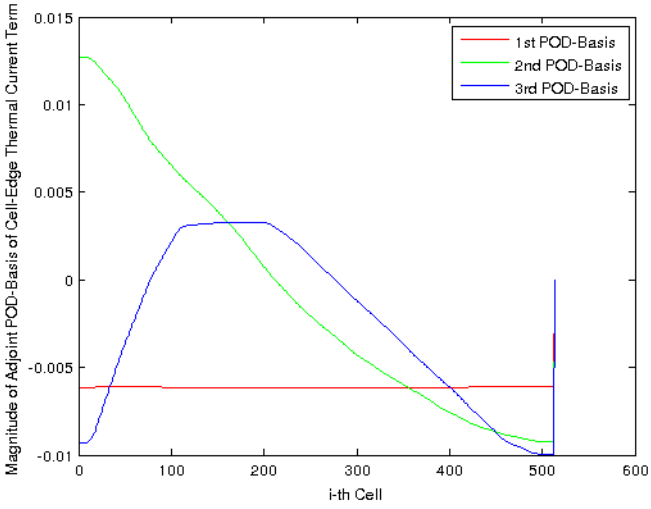

Figure 5.28: First Three POD-bases of Adjoint Solution of Thermal Current

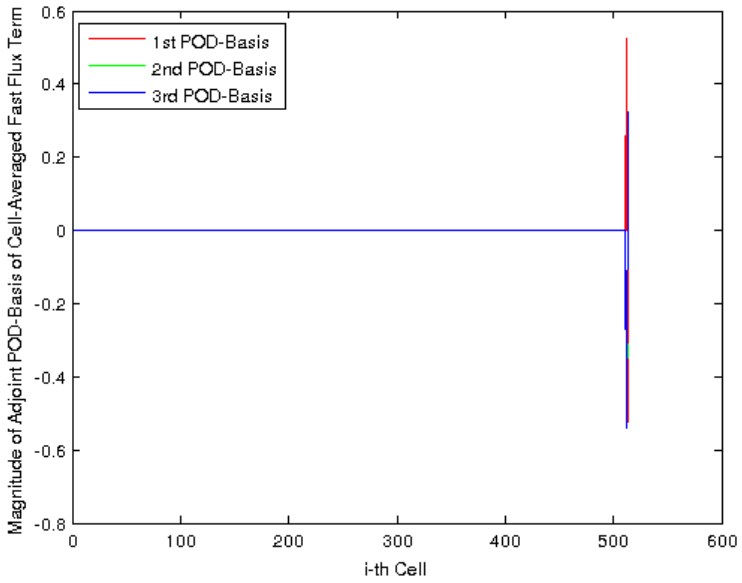

Figure 5.29: First Three POD-bases of Adjoint Solution of Fast Flux

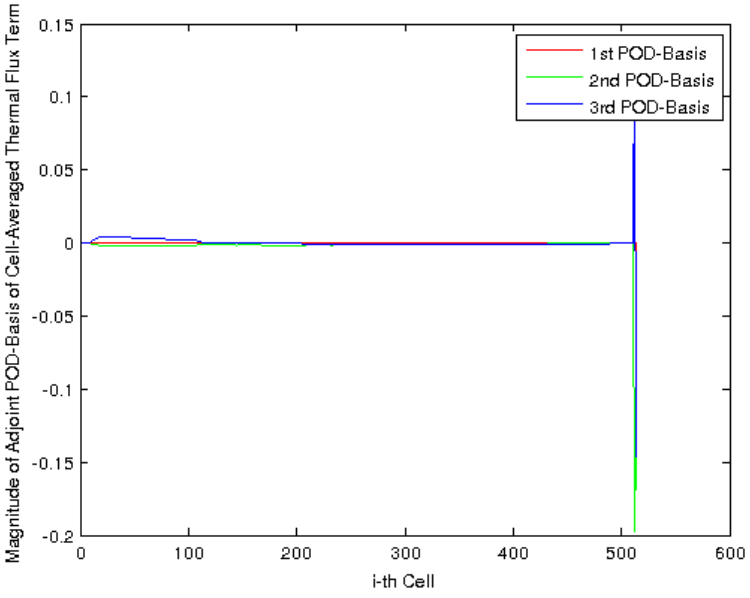

Figure 5.30: First Three POD-bases of Adjoint Solution of Thermal Flux

Fig. 5.31-5.32 show errors resulted from the POD methods defined by Eq.(3.4). One important note is that the actual errors in the POD-based methods do not match with Fig. 5.31-5.32. It is simply because these error estimates do not consider any effects of stochasticity in the original problem.

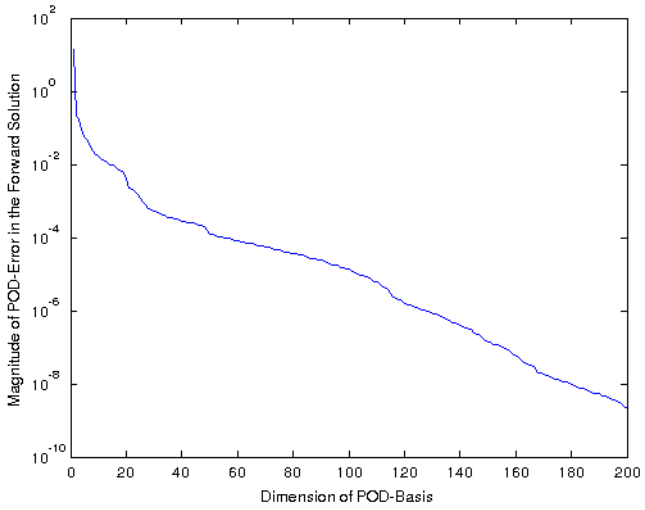

Figure 5.31: POD-Error in Eq.(3.4) for the Forward Solution

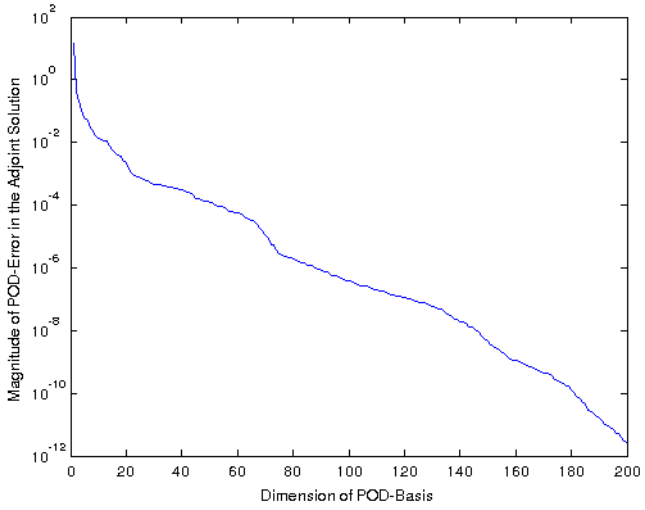

Figure 5.32: POD-Error in Eq.(3.4) for the Adjoint Solution

Now, these POD-bases are utilized to reduce the computational cost of the model B. To demonstrate the reduction, the mesh size is now refined to 1024. Since in the model B, the cell-edge current, the cell-edge flux, and the cell-averaged flux for thermal and fast group are computed together, the error is computed by the following equation:

$$\max_{j=1,\cdots,25} \left|\left| \begin{bmatrix} J_{edge}(\xi^j) \\ \phi_{edge}(\xi^j) \\ \phi_{average}(\xi^j) \end{bmatrix} - \begin{bmatrix} J^{\mathrm{POD}}_{edge}(\xi^j) \\ \phi^{\mathrm{POD}}_{edge}(\xi^j) \\ \phi^{\mathrm{POD}}_{average}(\xi^j) \end{bmatrix} \right|\right|_2 \tag{5.5}$$

where the eigenvector is normalized by:

$$\left|\left| \begin{bmatrix} J_{edge}(\xi^j) \\ \phi_{edge}(\xi^j) \\ \phi_{average}(\xi^j) \end{bmatrix} \right|\right|_2 = 1 \forall j. \tag{5.6}$$

The POD-Galerkin method, the adjoint approach ($\gamma = 0$), the adjoint approach ($\gamma = 0.0005$), and the k-LS approach are benchmarked against 25 realizations of the solution generated by randomly perturbing the macroscopic cross-section values. Fig. 5.33 shows the error obtained by Eq.(5.5). As expected, the adjoint approach ($\gamma = 0$) is highly unstable to recover the eigenvector compared with the model A. As discussed previously, this is due to the fact that the adjoint POD-basis has almost zero everywhere except a few points. On the other than, the adjoint approach ($\gamma = 0.0005$) shows slightly better results around $r = 150$. However, the adjoint approach ($\gamma = 0.0005$) is still unstable for recovering the eigenvector. On the other hand, the k-LS approach demonstrates almost monotonic convergence. This agrees with the intuition that as the number of the basis increases, the error decreases. The POD-Galerkin method shows a good result in this model even though it is not guaranteed to work with generalized eigenvalue problem.

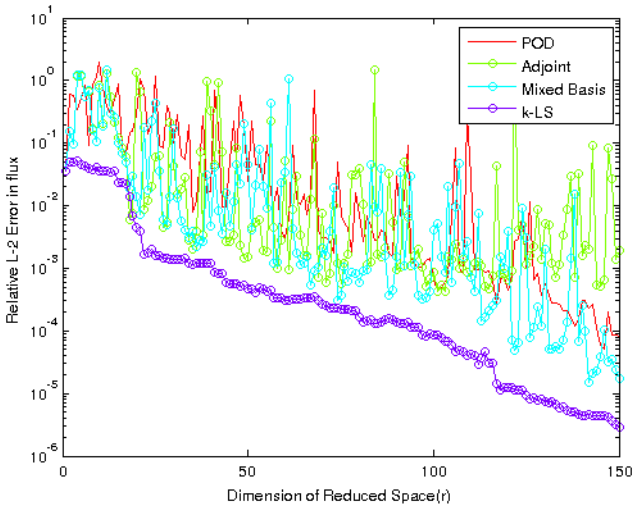

Figure 5.33: Comparison of the POD-Based Methods by Eq.(5.5)

Fig. 5.34 shows the relative error in the k-eigenvalue. The most significant observation is that the adjoint approach ($\gamma = 0$) shows an excellent convergent rate compared to other POD-based methods, even though it did a poor job with fluxes. Note that when $r \approx 80$, the adjoint approach ($\gamma = 0$) showed a steep peak error. This is due to the fact that Newton's mMthod failed to converge to the dominant eigenvalue of the original problem, but it instead converged to the newly introduced eigenvalue. If all eigenvalues are computed by the QZ algorithm, then one can find the appropriate eigenvalue easily. The reduced eigenvalue problem is small enough to use QZ algorithm to calculate all eigenvalues. The adjoint approach ($\gamma = 0.0005$) also shows good convergence. On the otehr hand, the POD-Galerkin method shows the worst results for the k-eigenvalue problem as expected. One interesting observation is that the k-LS approach showed almost monotonic convergence in the k-eigenvalue. Fig. 5.35 shows a performance study of each method when calculating the k-eigenvalue with the dimension of the POD-basis equal to 100. As expected, the adjoint approach($\gamma = 0$) shows the best performance compared to other methods and the adjoint approach ($\gamma = 0.0005$) is the second best. The POD-Galerkin projection shows the worst performance in this problem since the basis for the forward solution do not form the basis for the adjoint solution.

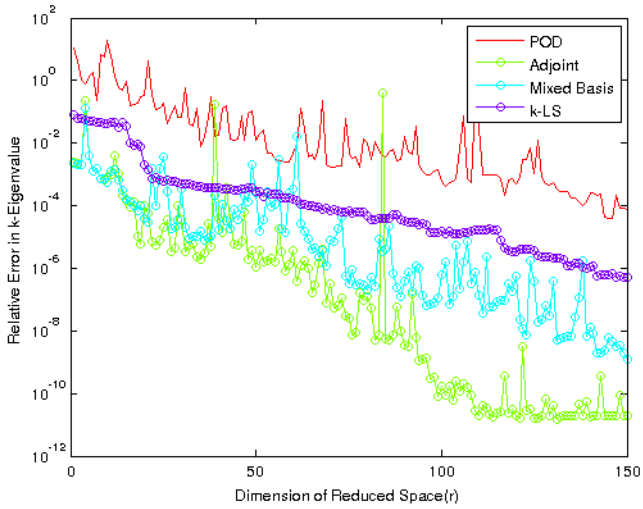

Figure 5.34: Comparison of the POD-based Methods using Relative Error in k-Eigenvalue

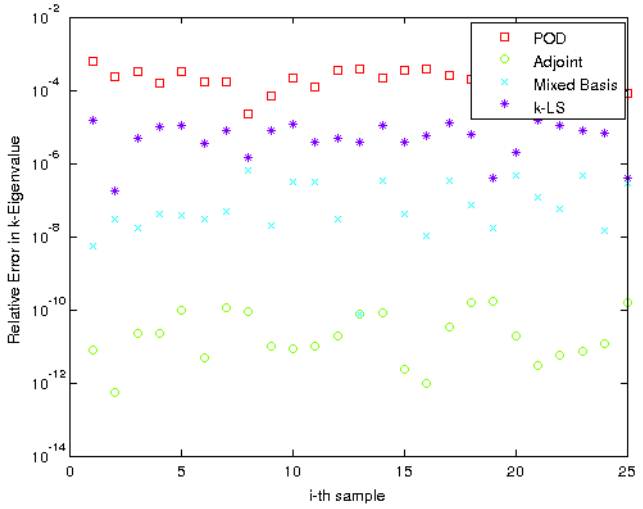

Figure 5.35: Comparison of the POD-based Methods by Relative Error in k-Eigenvalue at $r = 100$

Fig. 5.36-5.38 shows the L_2 norm errors in the eigenvector from the POD-based meth-

ods at $r = 50$, $r = 100$, and $r = 150$. Overall, the k-LS method showed the best performance compared to the other methods. This is due to the fact that the k-LS method does not rely on the adjoint solutions at all. The adjoint approach ($\gamma = 0.0005$) showed the second best performance at $r = 150$. However, it also showed poor performance at $r = 50$ and 100. The adjoint approach ($\gamma = 0$) and the POD-Galerkin method showed poor performance in the flux recovery.

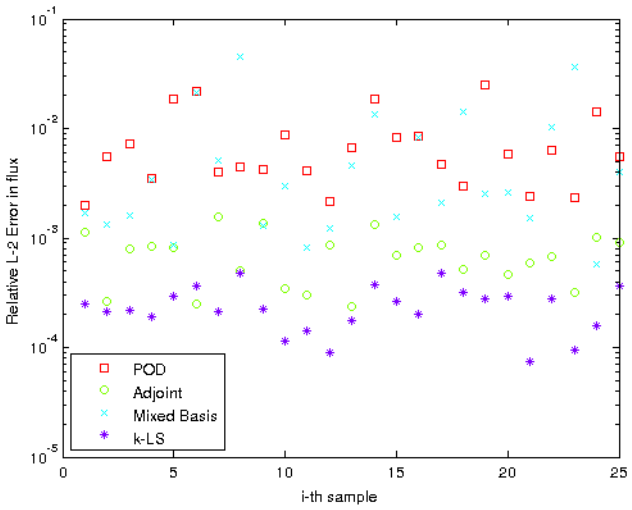

Figure 5.36: Comparison of the POD-based Methods using Eq.(5.5) at $r = 50$

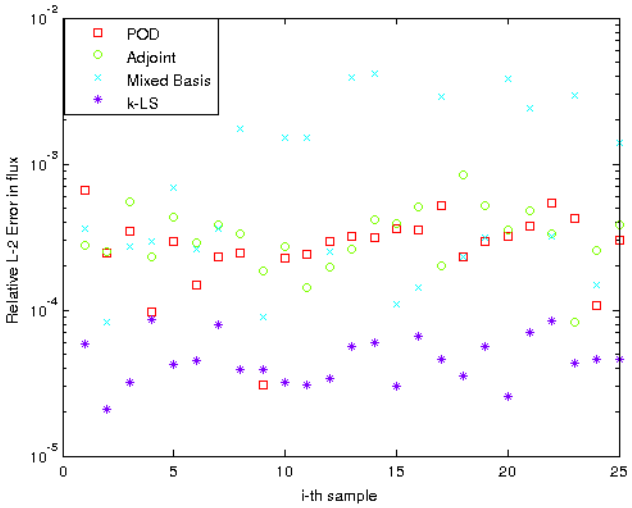

Figure 5.37: Comparison of the POD-based Methods using Eq.(5.5) at $r = 100$

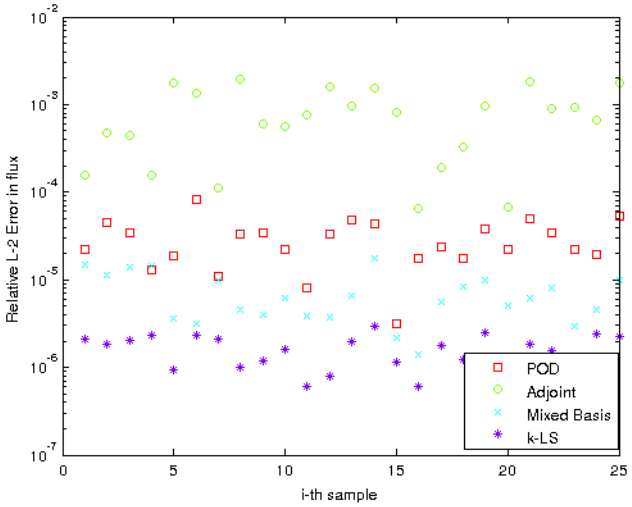

Figure 5.38: Comparison of the POD-Based Methods by Eq.(5.5) at $r = 150$

Test Model E

The same numerical test is applied to the test model E. In this model, each atomic density is assumed to be uniformly distributed within 10% of its reference value and each microscopic cross-section value is assumed to be uniformly distributed within 5% of its reference value. Therefore, the number of total input parameters is $37\times(44\times(44+2))=74888$.[1] Fig. 5.39 shows the singular value decomposition of 500 forward and adjoint solutions generated by random perturbations in atomic densities and microscopic cross-sections.

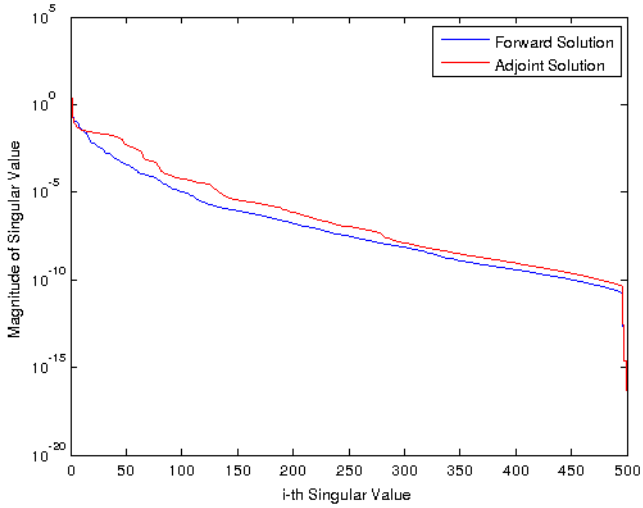

Figure 5.39: Singular Value Plot for Forward and Adjoint Solutions

Now, using these solutions as thePOD-bases, the 2D multi-group diffusion problem is reduced. Different from previous two numerical tests, the methods are benchmarked against only one realization of a randomly generated eigenpair for each r because of its high computational cost. The methods are also benchmarked against 300 realizations at selected dimension of r. In this model, the value of γ is chosen to be 0.8. Fig. 5.40-5.41 show a comparison of the POD-based methods described in Chapter 3. Fig. 5.40 is consistent with test model A but not with test model B. The k-LS method shows a slow convergence rate when compared to other methods. It is important to note that for flux

[1] 2D Scattering, Absorption, and Fission

calculations the traditional POD-Galerkin approach shows the best result. This is due to the fact that the basis for forward solutions forms the basis for the adjoint solutions in this problem.

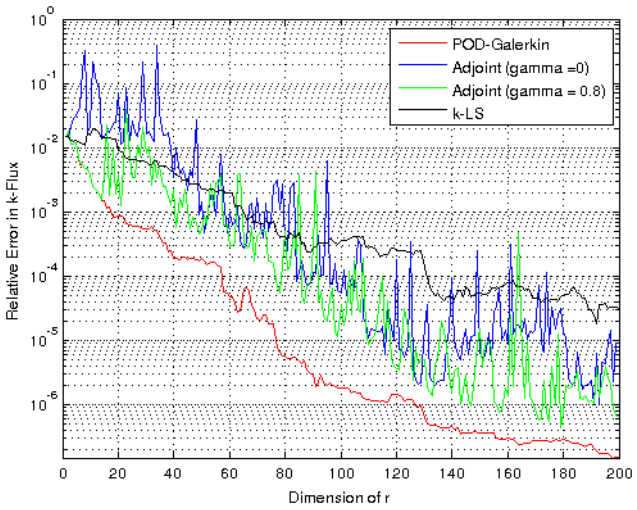

Figure 5.40: Comparison of POD-Based Method for the Scalar Flux

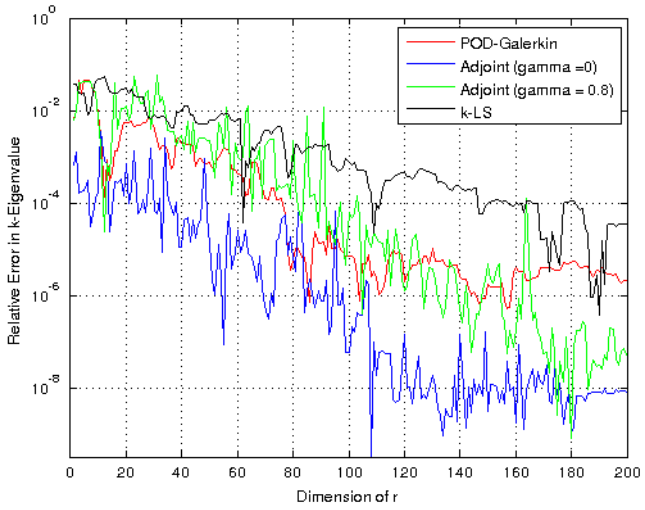

Figure 5.41: Comparison of POD-Based Method for the k-Eigenvalue

For the case of k-eigenvalue, the adjoint approach ($\gamma = 0$) shows the fastest convergence rate as expected. Based on Fig. 5.41, the effective rank is approximately 180, which is far less than the original dimension of the problem (74932). It is important to note that the majority of computational time is spent on the projection procedure of $\mathbf{L}$ and $\mathbf{F}$. The original system requires reordering techniques and Krylov-type iterative methods to be solved in reasonable amount of time. On the other hand, the reduced system no longer requires any sophisticated approaches to minimize the computational time.

Fig. 5.42-5.45 show the comparison of the POD-based methods with 300 realizations of the k-eigenvalue with randomly perturbed atomic densities and microscopic cross-sections at $r = 50, 100, 150$, and 200. Overall, the k-LS method shows the worst results compared to the other results. As expected, the adjoint approach ($\gamma = 0$) shows the best results and fastest convergence rate. The adjoint approach ($\gamma = 0.8$) shows second worst at $r = 50$ and 100. However, it gradually improves as r increases.

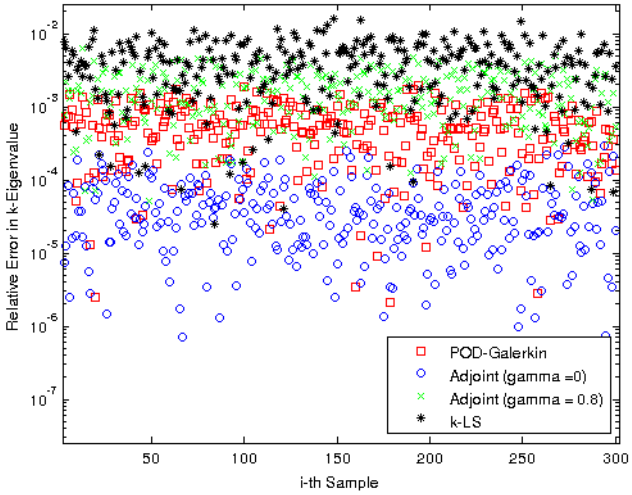

Figure 5.42: Comparison of POD-Based Method for the k-Eigenvalue at $r = 50$

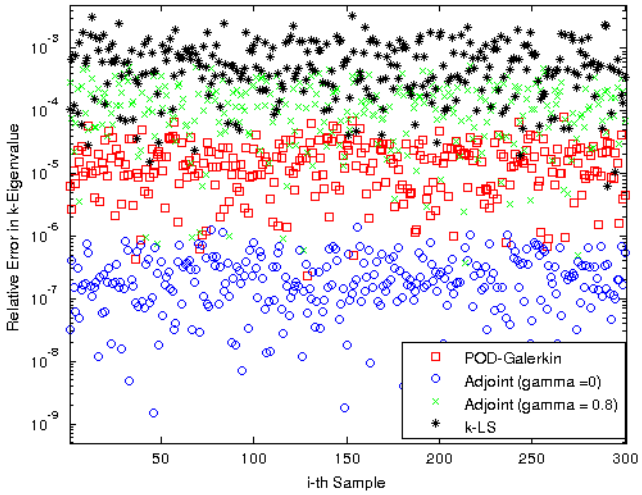

Figure 5.43: Comparison of POD-Based Method for the k-Eigenvalue at $r = 100$

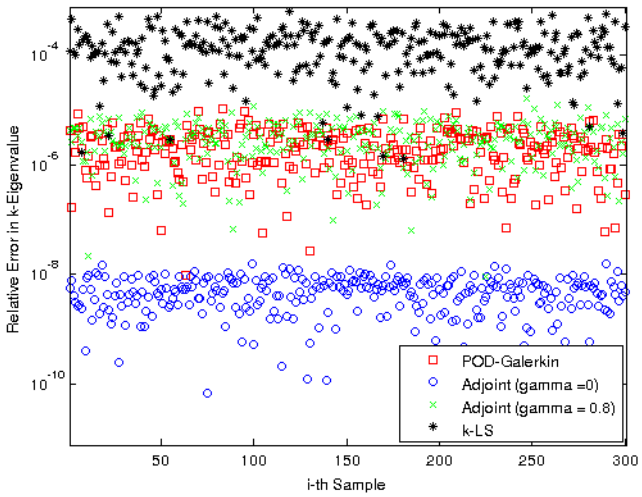

Figure 5.44: Comparison of POD-Based Method for the k-Eigenvalue at $r = 150$

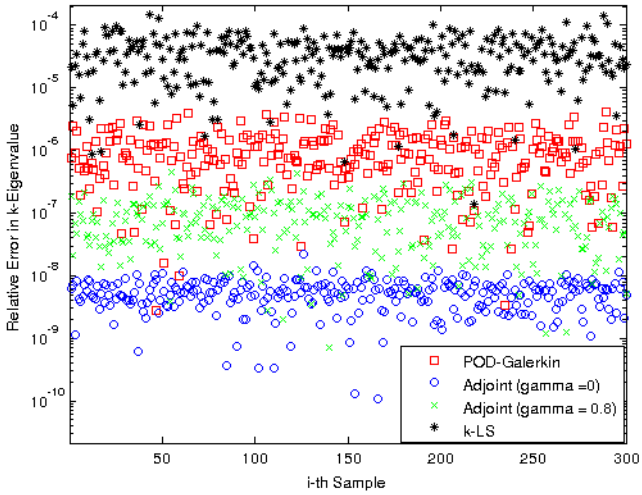

Figure 5.45: Comparison of POD-Based Method for the k-Eigenvalue at $r = 200$

Fig. 5.46-5.49 show the comparison of the POD-based methods with 300 realizations of fluxes at $r = 50, 100, 150$, and 200. Different from the k-eigenvalue, the POD-Galerin method shows the best results in L_2 norm errors. The adjoint approach ($\gamma = 0.8$) shows a large fluctuation in its error at $r = 100$. Other than that, the adjoint approach ($\gamma = 0.8$) gives the second best results. The adjoint approach ($\gamma = 0$) shows the second worst performance. In this model, the k-LS method shows consistently worst results for k-eigenvalues and fluxes.

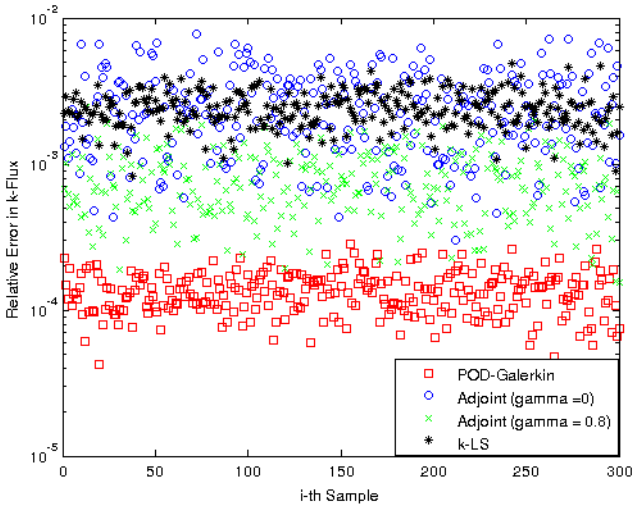

Figure 5.46: Comparison of POD-Based Method for the Flux at $r = 50$

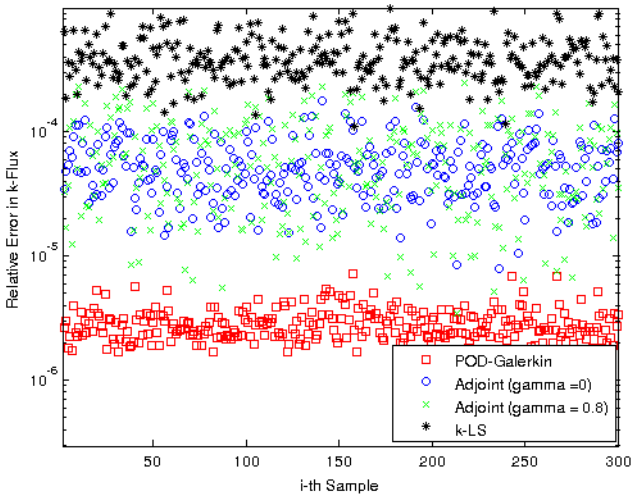

Figure 5.47: Comparison of POD-Based Method for the Flux at $r = 100$

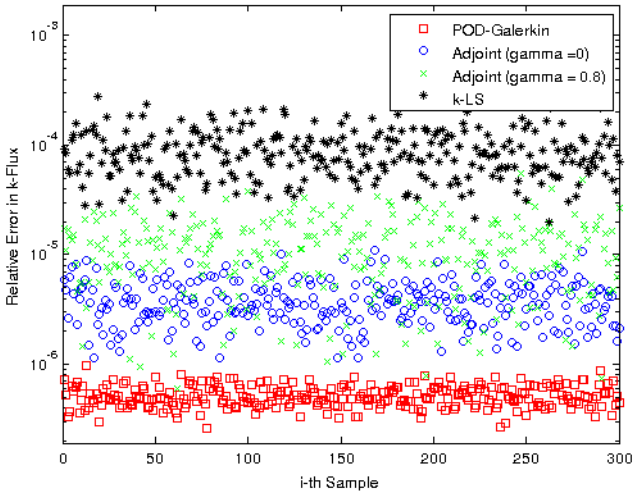

Figure 5.48: Comparison of POD-Based Method for the Flux at $r = 150$

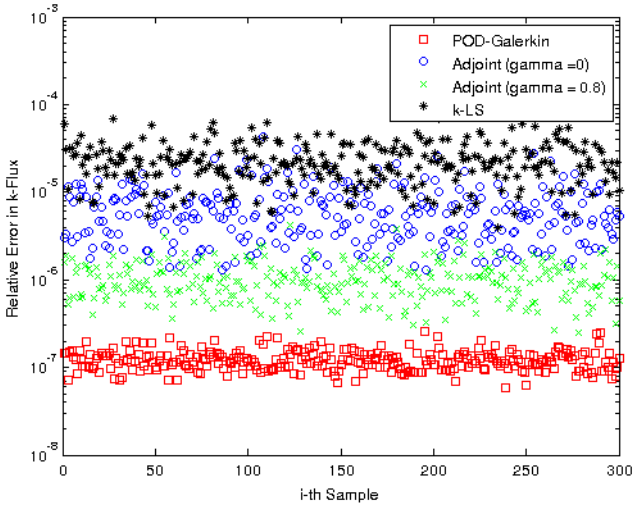

Figure 5.49: Comparison of POD-Based Method for the Flux at $r = 200$

Overall, POD-based methods can reduce the model order successfully. The adjoint approach ($\gamma = 0$) shows consistently the best results in the k-eigenvalue calculation, but its flux recovery is not accurate when compared to the POD-Galerkin method. Therefore,

it is natural to use the adjoint approach ($\gamma = 0$) for the k-eigenvalue calculations and to use the POD-Galerkin method for the flux calculations. Most of the computational cost in the state-level ROM lies in the projection procedure to generate the reduced order model. Once the reduced order matrices are generated, the generalized eigenvalue problem can be solved instantly. Therefore, solving both of the POD-Galerkin method and the adjoint approach ($\gamma = 0$) is the best option for us.

Chapter 6

Numerical Results for generalized Polynomial Chaos

First of all, Theorem 2 should be numerically verified. Assume that the Vandermonde-like matrix of orthonormal polynomials is constructed with Legendre polynomials and Chebyshev polynomials. According to Theorem 2, the probability distribution of the random number can be of any form. For simplicity, the uniform distribution is chosen for the Legendre polynomials and Eq.(4.62) is chosen for the Chebyshev polynomials. Fig. 6.1 shows the demonstration of Theorem 2 with 12 stochastic variables and the order of expansion is 5. The horizontal axis shows how many times the Vandermonde-like system is over-sampled. Note that the square Vandermonde-like system requires 6188 samples. As can be seen, the condition number decays quickly to the theoretical lower limit of 1. This agrees with Theorem 2.

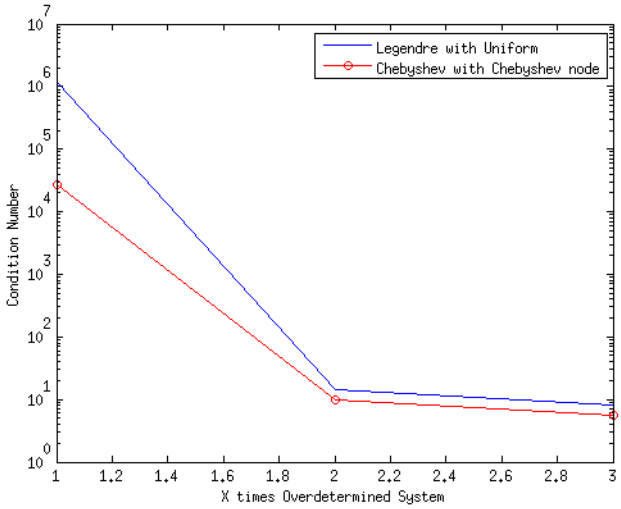

Figure 6.1: Condition Number of Legendre and Chebyshev Polynomials

Another important observation is that the condition number for the Chebyshev polynomials is smaller than that of Legendre polynomials. This also agrees with the fact that the Lebesgue constant for the Chebyshev polynomials is better than that of the Legendre polynomials. If the coefficients of the Chebyshev polynomials are known, then the coefficients of the Legendre polynomials can be easily obtained and vice versa. Therefore, without loss of generality, the Chebyshev polynomials can be used for performing non-intrusive polynomial chaos, and then the Legendre polynomials are recovered after finding the coefficients of the Chebyshev polynomials.

Now, test model A is evaluated using non-intrusive gPC. Each macroscopic cross-section is assumed to be uniformly distributed within $\pm$ 10% of the reference value. In this model, the Legendre polynomials are used. First of all, the state variables of model A are reduced by the POD-Galerkin method as described in Chapter 3. Then, the k-eigenvalue and forward solutions can be sampled less computational time than the original model. The accuracy of the POD-Galerkin method in test model A can also be found in Chapter 3. Fig. 6.2 shows the coefficients of the Chebyshev polynomials up to 5th order obtained by L_2 regularization with 12376 samples. It can be observed that the magnitude of the coefficients decays quickly and it only depends on a few of polynomials. Therefore, in this problem the sparse approximation is valid. Fig. 6.3 shows the comparison of the coefficients up to 4th-order obtained by L_1 regularization with

L_1 magic [37] and L_2 regularization. The sparse approximation successfully recovered the dominant coefficients. Even though L_1 optimization can recover the dominant coefficients with fewer samples, it cannot recover the remaining coefficients. To compare L_1 and L_2 regularization, 250 realizations of the k-eigenvalue are generated by random perturbation of the macroscopic cross-sections. Fig. 6.4 shows the relative errors in the k-eigenvalue with L_1 and L_2 regularization with the order of expansion equal to 4. Twice over-sampled Vandermonde-like system (n = 3640) showed that the best result is around 1 pcm. On the contrary, the square Vandermonde-like system (n = 1820) showed the worst results around 40 pcm of errors. L_1 regularization (n = 750) showed better results than the square Vandermonde-like system (n = 1820), but it is not as accurate as the over-determined Vandermonde-like system (n = 3640).

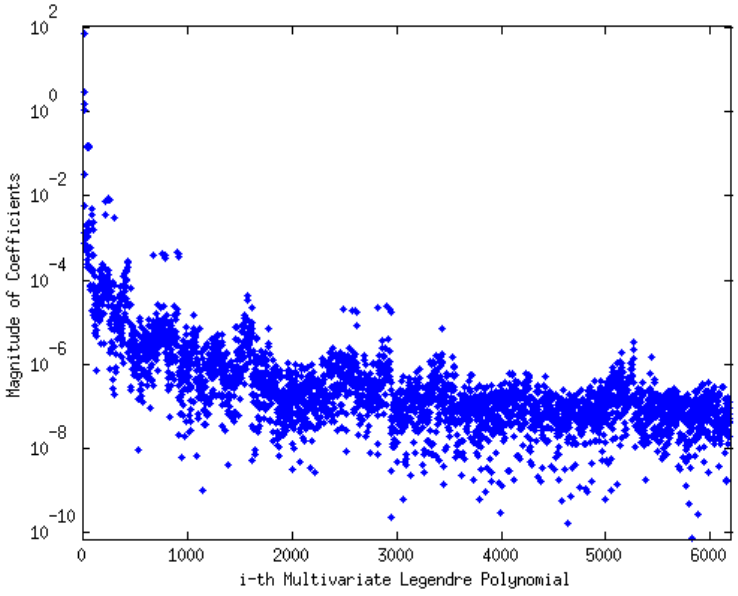

Figure 6.2: Coefficients of Legendre Polynomials obtained from L_2 regularization

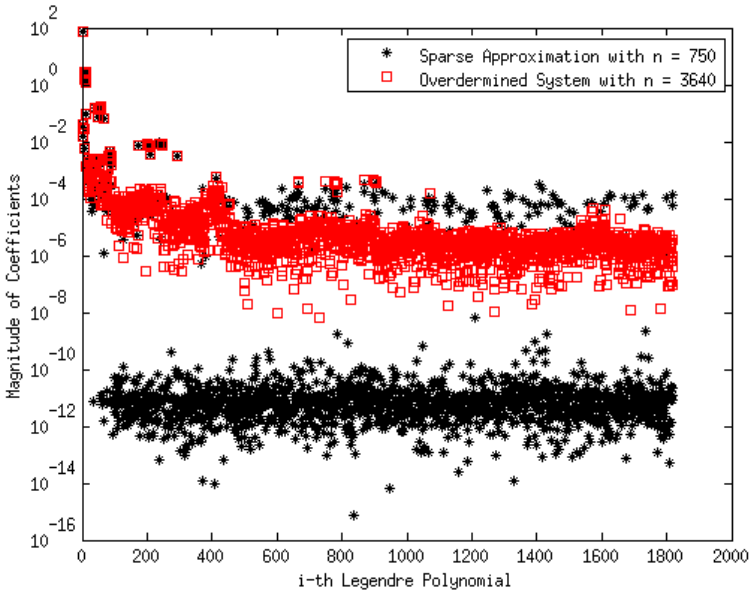

Figure 6.3: Comparison of Coefficients obtained from L_1 and L_2 Regularization Problem

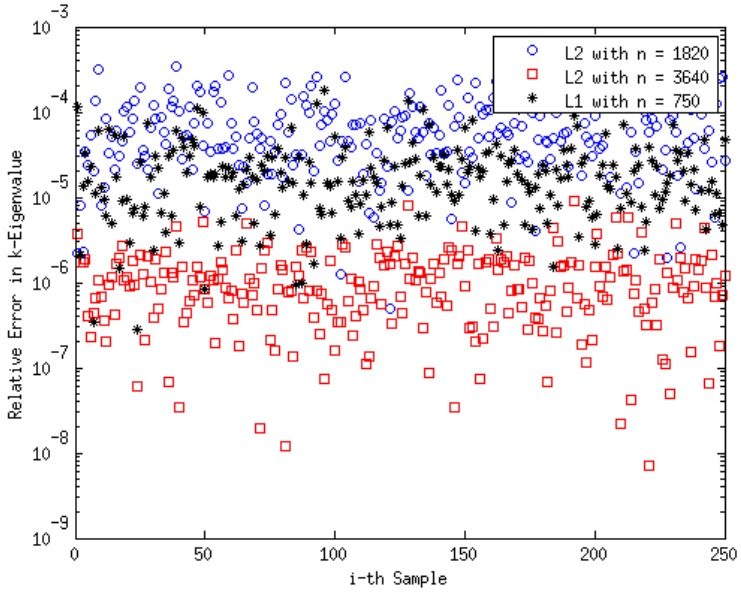

Figure 6.4: Comparison of Relative Error in k-eigenvalue with L_1 and L_2 Regularization

Fig. 6.5 shows the comparison of L_1 regularization with differing number of samples with the order of expansion up to 5. As observed in Fig. 6.5, the accuracy increases with the number of samples.

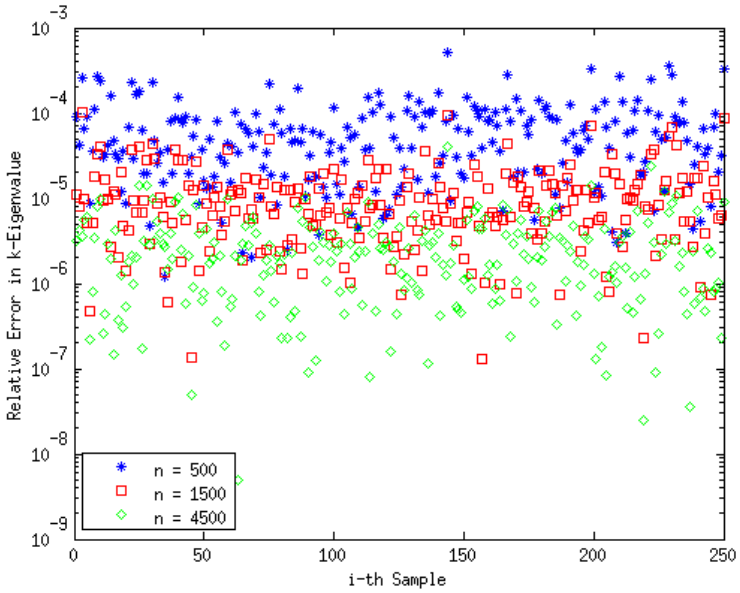

Figure 6.5: Comparison of L_1 Regularization with Different Number of Samples

Unfortunately, test model A is a very simple case and is not realistic. Therefore, the sparse approximation is applied to test model C.

In test model C, the microscopic cross-section is assumed to be uniformly distributed within $\pm$ 10% of its reference value. A 44 energy group cross-section is used and 8 isotopes, U-234, 235, 238, 239, Pu-238, 239, 240, and 241 are perturbed randomly. In addition, only three reaction types are considered. They are: fission, capture, and γ-n reactions. Therefore, the total number of input parameters is $44 \times 8 \times 3 = 1056$. Unfortunately, according to Eq.(4.5), the number of samples to recover the coefficients of the chaos expansion by the non-intrusive approach is enormous. Therefore, in this model, instead of state-level reduced-order modeling, the input parameter reduction is applied. First, the sensitivity of the model C is sampled by randomly perturbing the cross-section values using the adjoint method. Note that these adjoint solutions can also be used for

model order reduction as well. SAMS from SCALE automatically calculate the relative sensitivity as:

$$S_{\Sigma_r^g}^{\text{relative}} = \frac{\Sigma_r^g}{k}\frac{\partial k}{\partial \Sigma_r^g} \tag{6.1}$$

To obtain the sensitivity with respect to the input parameters, the following equation is used:

$$S_{\xi_i} = \frac{\partial k}{\partial \xi_i} = \sum_{g=1}^{G}\sum_{i=1}^{R}\frac{\partial k}{\partial \Sigma_i^g}\frac{\partial \Sigma_i^g}{\partial \xi_i} \tag{6.2}$$

where R is the number of macroscopic cross-sections, i is the reaction type, g is the energy group index, and ξ_i is i-th component of the stochastic vector. Then, the sensitivity matrix can be constructed in terms of the stochastic variables instead of cross-sections.

Using the sensitivity information obtained the input parameter space is reduced. Fig. 6.6 shows the errors of the input parameter reduction calculated by the following equation,

$$\max_{j=1,\cdots 25}\frac{\left|k(\xi_j) - k(\mathbf{U}_r\mathbf{U}_r^T\xi_j)\right|}{k(\xi_j)}. \tag{6.3}$$

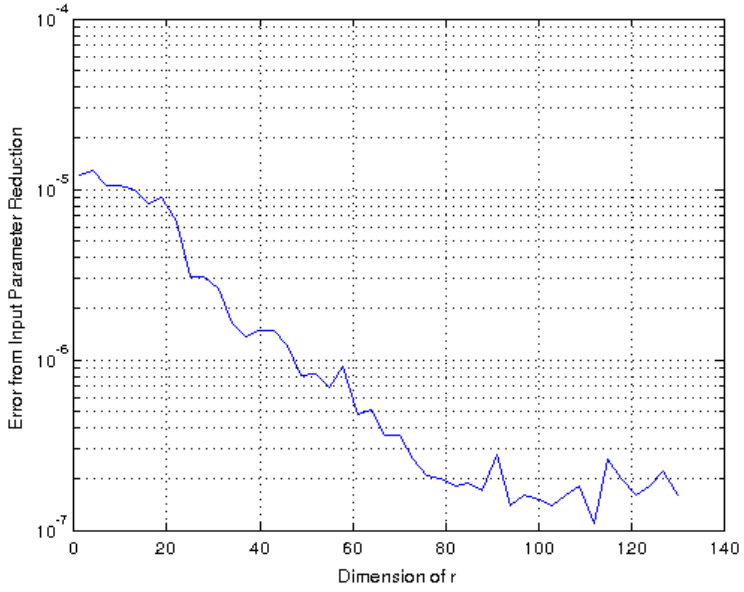

Figure 6.6: Input Parameter Reduction of the Pin-Cell Model

Why didthe input parameter reduction work so well in this case? It is because the k-eigenvalue is almost a *linear* function with respect to the input parameter space in this model with this specific perturbation. In other words, there is only one important direction in the input parameter space. Hence, large input parameter reduction is possible for this problem. It is important to note that many models are not linear at all. In such cases, one cannot expect this degree of reduction. The non-linear case is demonstrated with the model E.

Using input parameter reduction, the k-eigenvalue and scalar fluxes can be recovered in terms of the reduced input parameter space. Note that when the original input parameter is projected onto the reduced space, it does not preserve the probability distribution of the original input parameters, and gPC is no longer applicable. Instead, polynomial interpolation is used with orthogonal polynomials. From Fig. 6.6, in order to achieve 1pcm the dimension of $r = 11$ is required. Now, the k-eigenvalue and scalar flux are represented by the reduced input parameter with

$$k(\alpha) = \sum_{i=0}^{M} k_i P_i(\alpha) \tag{6.4}$$

and

$$\phi(\alpha) = \sum_{i=0}^{M} \phi_i(x) P_i(\alpha) \tag{6.5}$$

where $\phi_i(x)$ can be represented in the POD-basis such that

$$\phi_i(x) = \sum_{j=1}^{N} \phi_{ij} \Phi_j(x) \tag{6.6}$$

Note that the POD-basis, $\Phi_j(x)$, is obtained at the same time as the sensitivity matrix is computed. Even though the POD-basis is not used to reduce the dimension of the model, it can still reduce the computational cost of the interpolation tremendously as discussed in Chapter 3. In this problem N is chosen to be 20. In order to minimize the number of polynomials, multivariate Chebyshev polynomials are chosen for $P_i(\alpha)$ and the Orthogonal Marching Pursuit Algorithm is used instead of the L_1 regularization. The reason behind using the Orthogonal Marching Pursuit method is that the computational cost of the L_1 regularization increases rapidly as the dimension of the Vandermonde-like

matrix increases. Even though the L_1 regularization can typically find a more accurate solution with less number of samples, the Orthogonal Marching Pursuit algorithm will require less computational time. Fig. 6.7-6.10 show the coefficients of the Chebyshev polynomials for the k-eigenvalue and scalar flux recovered by L_2 regularization and the Orthogonal Marching Pursuit algorithm. It can be seen that the Orthogonal Marching Pursuit algorithm successfully recovers the dominant coefficients.

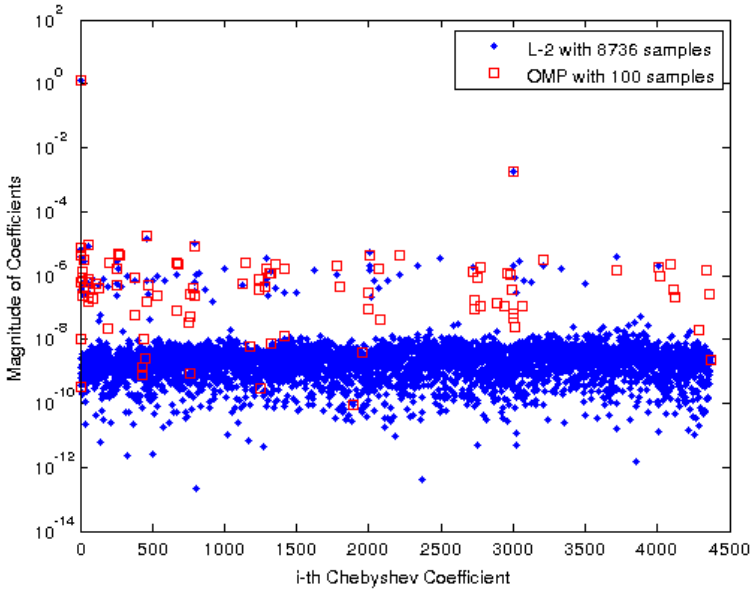

Figure 6.7: Coefficients of k-eigenvalue Interpolation recovered by OMP and L_2 Regularization

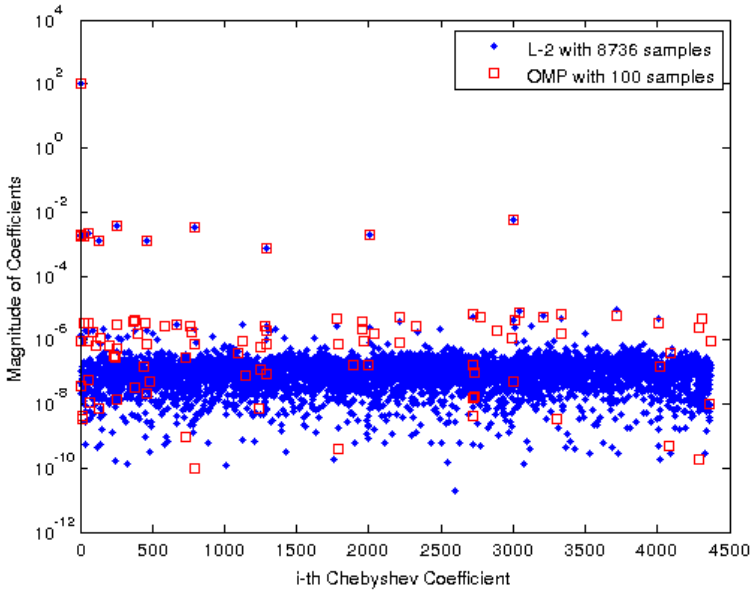

Figure 6.8: Coefficients of 1st POD-basis, ϕ_{1j}, recovered by OMP and L_2 Regularization

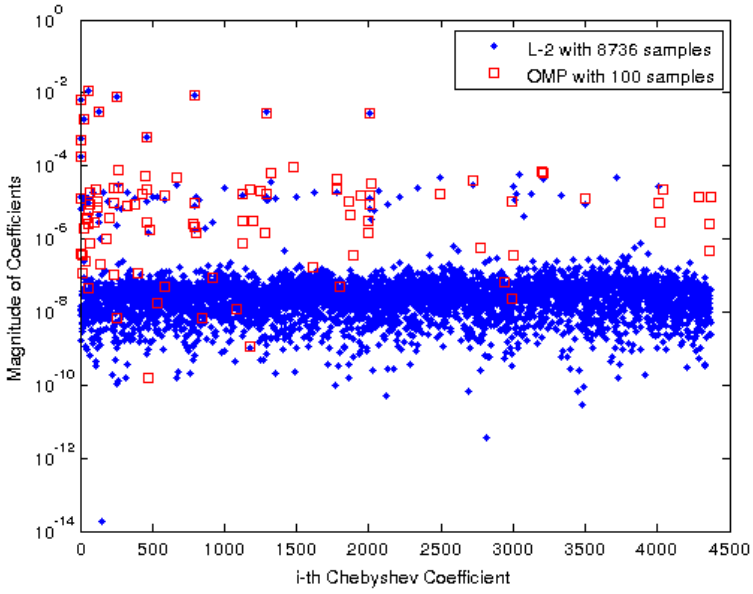

Figure 6.9: Coefficients of 2nd POD-basis, ϕ_{2j}, recovered by OMP and L_2 Regularization

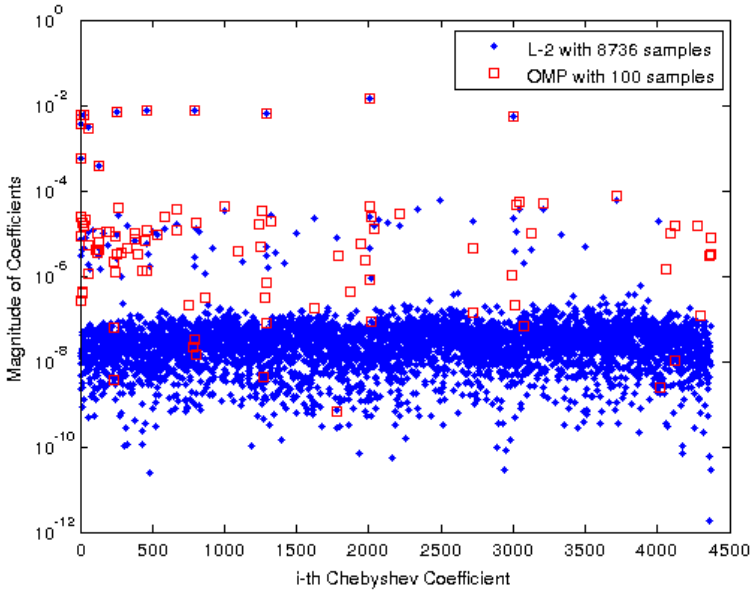

Figure 6.10: Coefficients of 3rd POD-basis, ϕ_{3j}, recovered by OMP and L_2 Regularization

Now, the recovered coefficients for the k-eigenvalue and scalar flux are validated by 500 realizations of the k-eigenvalues and fluxes generated by random perturbation of cross-section values. Fig. 6.11-6.12 show the relative errors in the k-eigenvalue and flux in the terms of L_2 norm.

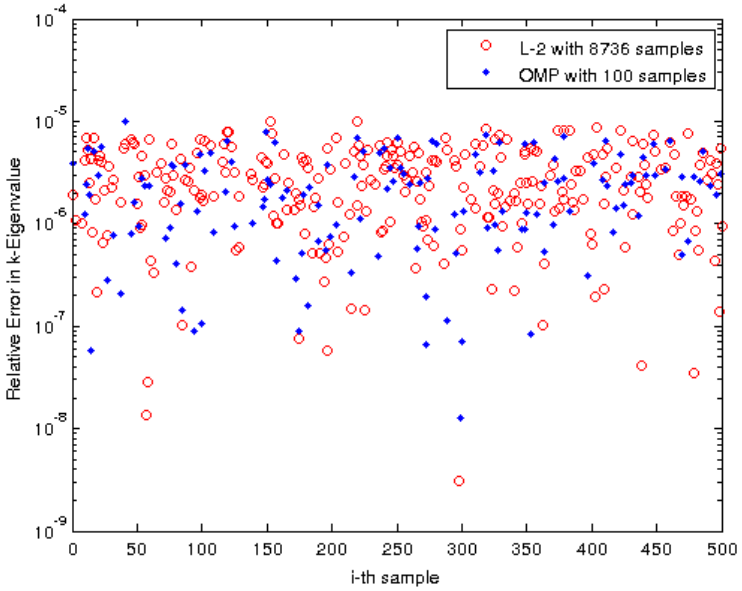

Figure 6.11: Relative Error in k-Eigenvalue with L_2 Regularization and OMP

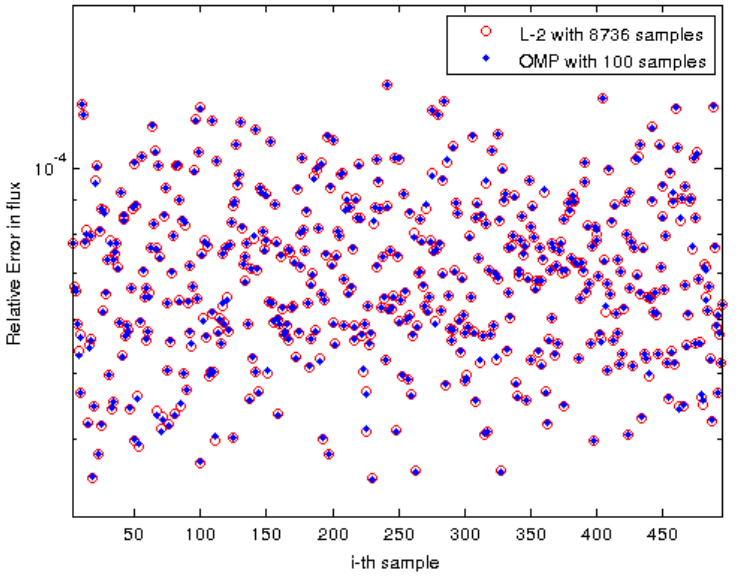

Figure 6.12: Relative L_2 Error in Flux with L_2 Regularization and OMP

The Orthogonal Marching Pursuit and the input parameter reduction demonstrated excellent results in test case D. However, these approaches are not applicable for large systems such as an assembly model.

To see the applicability of these methods, test model D, which models a quarter-assembly, is now evaluated using these approaches. In the test model D, each microscopic cross-section value is assumed to be uniformly distributed within $\pm$ 5% of its reference values. 44 energy group cross-sections are used and 8 isotopes (U-234, 235, 238, 239, Pu-238, 239, 240, and 241) are perturbed randomly. In addition, only three reaction types are considered. They are: fission, capture, and $\bar{\nu}$. Note that $\bar{\nu}$ is perturbed instead of (γ, n) reaction. Hence, the total number of input parameters is 1056. Now, the input parameter reduction is applied to this model. Fig. 6.13 shows the worst errors obtained from 25 realizations of solutions with input parameter reduction. In order to get 1 pcm, it is apparent that more than 400 pseudo-input parameters are required. Even though the model is still highly linear, the non-linearity plays a significant roles in the number of the pseudo-input parameters to obtain 1 pcm errros. This number of input parameters is computationally prohibitive even with the model order reduction scheme described in Chapter 3. Fig. 6.14 shows the required number of samples required by the L_2 regularization with a 5th order expansion. Even though L_1 regularization and the Orthogonal Marching Pursuit do not require as many samples as L_2 regularization shown in Fig. 6.14, we do not have any tools to predict the number of samples needed for L_1 regularization and the Orthogonal Marching Pursuit. That is, the sparsity of the problem cannot be determined unless we obtain the solution by L_2 regularization. Even when one tries to apply the Orthogonal Marching Pursuit algorithm to this problem, the number of columns is too large and a greedy algorithm like the Orthogonal Marching Pursuit algorithm becomes computationally intractable. Therefore, application of the sparse approximation results in computational costs that are unreasonable for a small problem like this. Therefore, generalized Polynomial Chaos expansion is in general not applicable to reactor simulation even with state-level reduction, input parameter reduction and sparse approximation.

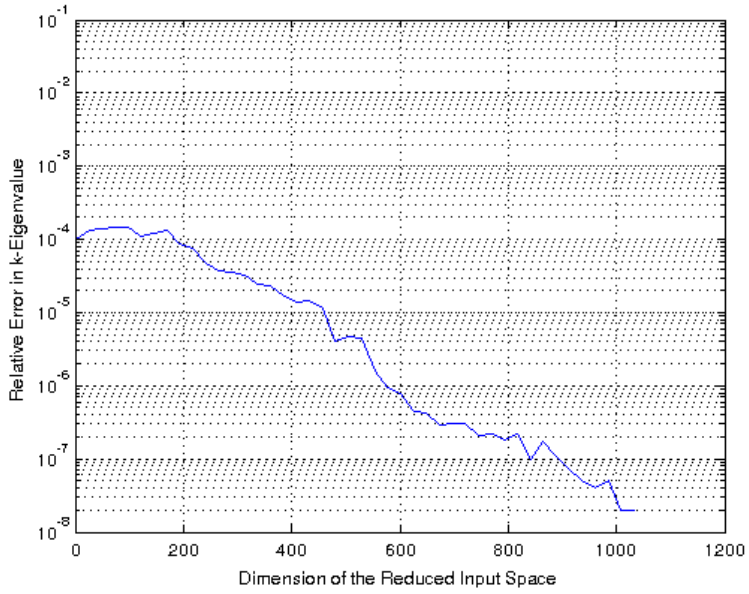

Figure 6.13: Error in Input Parameter Reduction

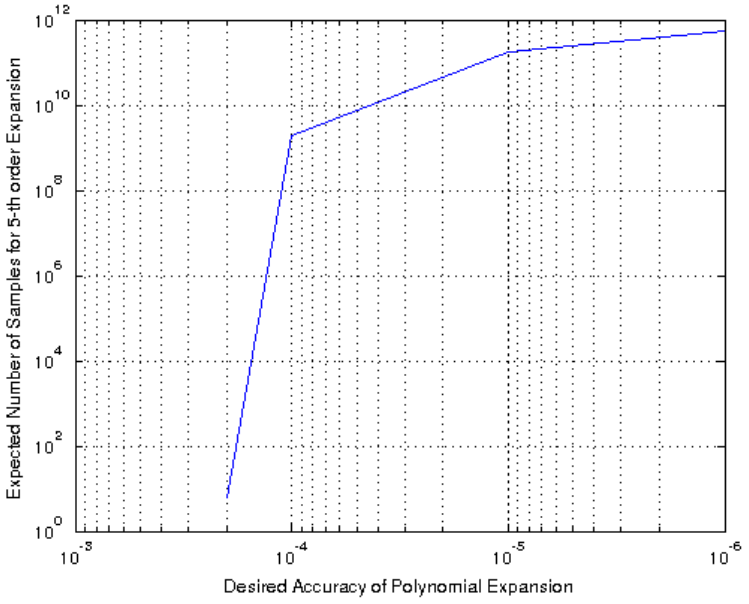

Figure 6.14: Estimated Number of Samples Required to achieve desired accuracy by L_2 Regularization with 5th Order Expansion

Chapter 7

Conclusions & Outlook

7.1 Conclusion

This thesis has presented new approaches to reduce a generalized eigenvalue problem in order to perform Uncertainty Quantification in the context of large scale nuclear reactor simulation. The POD-based methods described in this thesis can be applied to any generalized eigenvalue problem encountered in various engineering disciplines as long as the solution to the generalized eigenvalue problem has a low numerical rank. The numerical rank of the model can be estimated by the Proper Orthogonal Decomposition and the model is reduced by the forward POD-basis and adjoint POD-basis. Then, the generalized Polynomial Chaos method can be applied to the reduced order form of the diffusion model with the sparse approximation. It demonstrated that the number of samples required can be reduced by the sparse approximation. In addition, the input parameter reduction is applied to realistic transport models with the polynomial interpolation. The input parameter reduction can reduce the number of input parameters when the model is highly linear. However, the curse of dimensionality played significant role in the polynomial interpolation even after the input parameter reduction. In a realistic neutronics model, the non-intrusive generalized Polynomial Chaos method and polynomial interpolation are not applicable to constructing the surrogate models.

7.2 Future Work Recommendations

Future work with these POD-based methods will include the application to the neutron transport problem. The methods described can be applied to any neutronics codes, ranging from the diffusion approximation to the transport equation. Potential codes include NEWT, a component of the SCALE package from Oak Ridge National Laboratory, NESTLE, and any proprietary code uses the neutronics models. It also could be applied to the eigenvalue problem in structural engineering problems.

One interesting application of state-level reduction and input parameter reduction is to the dimension-adaptive sparse grid approximation. The POD-based methods can reduce the computational time required in the sparse grid approximation. Input parameter reduction can reduce the number of input parameters into hundreds or several thousands that could be dealt with the dimension-adaptive sparse grid approximation. Combining these subspace methods can make the sparse grid approach more effective for UQ. No method has so far successfully demonstrated Uncertainty Quantification of large scale high fidelity reactor model, but this approach may enable to demonstrate UQ of such systems.

REFERENCES

[1] J.M. BOOKER, T.J. ROSS, "An evolution of uncertainty assessment and quantification," *Scientia Iranica*, **18**, *3*, 669-676, (2011)

[2] FORNBERG, BENGT. *A practical guide to pseudospectral methods*, Cambridge University Press, 1996

[3] FINLAYSON, BRUCE A. *The method of weighted residuals and variational principles, with application in fluid mechanics, heat and mass transfer*, Academic Press, 1972

[4] D.M. LUCHTENBURG, B.R. NOACK, M. SCHLEGEL, "An Introduction to the POD Galerkin method for Fluid Flows with Analytical Examples and MATLAB source codes", Technical Report 01/2009, Berlin Institute of Technology (2009).

[5] K. KUNISCH, S. VOLKWEIN, "Galerkin Proper Orthogonal Decomposition Methods for a General Equation in Fluid Dynamics", *SIAM Journal on Numerical Analysis*, **40**, *2*, 492 (2003).

[6] HEATH, M. T., *Scientific Computing: An Introductory Survey, second ed.*, McGraw-Hill, (2002)

[7] SAAD, Y. *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, (1992)

[8] J. J. DUDERSTADT, L. J. HAMILTON, *Nuclear Reactor Analysis*, Wiley, (1976).

[9] STACEY, WESTON M. *Nuclear Reactor Physics*, Wiley-VCH, 2007

[10] E.L. WACHSPRESS, *Iterative Solution of Elliptic Systems*, Prentice-Hall, Englewood Cliffs, NJ, USA (1966).

[11] T. M. SUTTON, "Wielandt Iteration as Applied to the Nodal Expansion Method," *Nuclear Science and Engineering*, **98**, 169, (1988).

[12] D. R. FERGUSON, K. L. DERSTINE, "Optimized Iteration Strategies and Data Management Considerations for Fast Reactor Finite Difference Diffusion Theory Codes," *Nuclear Science and Engineering*," **64**, *2*, 593, (1977).

[13] D. F. GILL and Y. Y. AZMY, "A Jacobian-Free Newton-Krylov Iterative Scheme for Criticality Calculations Based on the Neutron Diffusion Equation," *in International Conference on Mathematics, Computational Methods, and Reactor Physics*, Saratoga Springs, NY, United States, (2009)

[14] D. F. GILL, "Newton-Krylov Methods for the Solution of the k-Eigenvalue Problem in Multigroup Neutronics Calculations," Doctoral dissertation, (2009).

[15] D. Y. ANISTRATOV, "Consistent Spatial Approximation of the Low-Order Quasi-Diffusion Equation on Coarse Grids", *Nuclear Science and Engineering*, **149**, *2*, 138, (2005).

[16] G. J. O'CONNOR, P. H. LIEN, "Burn-up Credit Criticality Benchmark - Phase IV-B: Results and Analysis of MOX Fuel Depletion Calculations", *Nuclear Energy Agency - Organisation for Economic Cooperation and Development*, Paris, France (2003)

[17] H. J. BUNGARTZ, M. GRIEBEL. "Sparse Grids," *Acta Numer.*, **13**, 147, (2004).

[18] Y. BANG, J. M. HITE, H. S. ABDEL-KHALIK, "Hybrid Reduced Order Modeling Applied to Nonlinear Models", *International Journal for Numerical Methods in Engineering,* DOI:10.1002/nme.4298, (2012).

[19] T. BUI-THANH, K. WILLCOX, O. GHATTAS, "Model Reduction for Large-Scale Systems with High-Dimensional Parametric Input Space", SIAM Journal on Scientific Computing, **30**, *6*, 3270, (2008).

[20] S. SOGA, H. ABDEL-KHALIK, "On Efficient Surrogate Model Construction for the Criticality Problem," in *Transactions of the American Nuclear Society*, Accepted (2012)

[21] K. KUNISCH and S. VOLKWEIN, "Galerkin Proper Orthogonal Decomposition Methods for a General Equation in Fluid Dynamics," *SIAM Journal on Numerical Analysis*, **40**, *2*, 492, (2003)

[22] S. VOLKWEIN, "Proper Orthogonal Decomposition and Singular Value Decomposition," Spezialforschungsbereich F003 Optimierung und Kontrolle, Projektbereich Kontinuierliche Optimierung und Kontrolle, Bericht Nr. 153, Graz, 1999.

[23] N. HALKO, P. MARTINSSON, J. TROPP, "Finding Structure Within Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions," Arxiv preprint arXiv:0909.4061 (2009).

[24] L. SIROVISH, "Turbulence and the Dynamics of Coherent Structures. Part 1 : Coherent Structures", *Quarterly of Applied Mathematics*, **45**, *3*, 561 (1987).

[25] I. C. F. IPSEN, *Numerical matrix analysis: linear systems and least squares*, SIAM, 2009

[26] G. BOUTRY, M. ELAD, G.H. GOLUD, P. MILANFAR, "The generalized eigenvalue problem for nonsquare pencils using a minimal perturbation approach," *SIAM Journal on Matrix Analysis and Applications*, **27**, *2*, 582, (2006).

[27] H. WOŹNIAKOWSKI. *Monte Carlo and Quasi-Monte Carlo Methods 2010*, Springerlinks, 2012

[28] T. GERSTNER, M. GRIEBEL, "Dimension-Adaptive Tensor-Product Quadrature," *Computing*, **71**, *1*, 65, (2003).

[29] M. S. ELDRED, J. BURKARDT, "Comparison of Non-Intrusive Polynomial Chaos and Stochastic Collocation Methods for Uncertainty Quantification," AIAA 2009-0976, (2009).

[30] M. M. R. Williams, "A Method for solving Stochastic Eigenvalue Problems," *Applied Mathematics and Computation*, **215**, *11*, 3906 (2010)

[31] R. ASKEY, J.WILSON, "Some Basic Hypergeometric Polynomials that Generalize Jacobi Polynomials," *Memoirs of the American Mathematical Society*,**319**, (1985)

[32] S. HOSDER, R. W. WALTERS, M. BALCH, "Efficient Sampling for Non-Intrusive Polynomial Chaos Applications with Multiple Uncertain Input Variables," *Proc. of the 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, No. AIAA-2007-1939, Honolulu, HI, April 2326, (2007).

[33] M. ELAD, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, Springer, New York (2010).

[34] A. DOOSTAN, H. OWHADI, "A Non-Adapted Sparse Approximation of PDEs with Stochastic Inputs," *Journal of Computational Physics*, **230**, *8*, 3015, (2011).

[35] E.J. CANDÉS, T. TAO, "The Dantzig Selector: Statistical Estimation When p Is Much Larger than n," *The Annals of Statistics*, **35**, *6*, 2313 (2007)

[36] S. BOYD, L. VANDENBERGHE, *Convex Optimization*, Cambridge University Press, (2004).

[37] E.J. CANDÉS, J. ROMBERG, "l_1-MAGIC: Recovery of Sparse Signals via Convex Programming," Tech. Rep., Caltech (2005).

[38] Y. S. BANG, H. S. ABDEL-KHALIK, J. M. HITE, "Hybrid Reduced Order Modeling Applied to Nonlinear Models", *International Journal for Numerical Methods in Engineering*, **91**, *9*, 929, (2012a).

APPENDIX

Appendix A

t-newt input files

A.1 Input File for MOX Pin-Cell Model

The following code listing is for a fresh MOX fuel pincell by T-NEWT sequence.

```
=tsunami-2d
MOX Unit Cell
xn44
read comp
 u-234        1 0 2.7999e-07 300    end
 u-235        1 0 5.857e-05 300    end
 u-238        1 0 0.023074 300    end
 pu-238       1 0 2.47e-05 300    end
 pu-239       1 0 0.00080623 300    end
 pu-240       1 0 0.00031298 300    end
 pu-241       1 0 0.00016533 300    end
 pu-242       1 0 5.3981e-05 300    end
 o-16         1 0 0.048992 300    end
 zr           2 0 0.042982 300    end
 fe           2 0 0.00014838 300    end
 cr           2 0 7.5891e-05 300    end
 h            3 0 0.066724 300    end
 o            3 0 0.033362 300    end
end comp
read celldata
  latticecell squarepitch  pitch=   1.33   3
                           fuelr=   0.412   1
                           cladr=   0.475   2 end
```

```
end celldata
read sams
nohtml
end sams
read model
MOX Unit Cell
read parm
  echo=yes saveangflx=yes
  epsinner= 1.00E-07
  epsthrm=  1.00E-07
  epsouter= 1.00E-07
  epseigen= 1.00E-07
  gptepsouter= 1.00D-07
  prtxsec=        yes
  sn=4
end parm
read materials
  mix=1 pn=1  end
  mix=2 pn=1  end
  mix=3 pn=1  end
end materials
read geometry
global unit 1
com="fresh"
 cuboid 1    0.665   0    0.665   0
 cylinder 2    0.412  sides=5
 cylinder 3    0.475  sides=5
 media 1 1 2
 media 2 1 -2 3
 media 3 1 1 -3
 boundary 1 5 5
end geometry
read bounds
  all=refl
end bounds
end model
end
=shell
cp ft42f001 ${RTNDIR}/ft42f001
cp ft92f001 ${RTNDIR}/ft92f001
```

```
cp senlib.sen ${RTNDIR}/senlib.sen
cp bonamist.sen ${RTNDIR}/bonamist.sen
cp worf ${RTNDIR}/worf
cp i_worker0002 ${RTNDIR}/i_worker0002
cp ft31f001 ${RTNDIR}/ft31f001
end
```

Appendix–A/input_scale.inp

A.2 Input File for Quarter Assembly Model

The following code listing is for a quarter assembly model.

```
=tsunami-2d
1/4 assembly model
xn44
read comp
  u-234        1 0 2.7999e-07 579    end
  u-235        1 0 5.857e-05 579    end
  u-238        1 0 0.023074 579    end
  pu-238       1 0 2.47e-05 579    end
  pu-239       1 0 0.00080623 579    end
  pu-240       1 0 0.00031298 579    end
  pu-241       1 0 0.00016533 579    end
  pu-242       1 0 5.3981e-05 579    end
  o-16         1 0 0.048992 579    end
  zirc2     2 1 579 end
  h2o       3 den=0.7135 1       579 end
  boron     3 den=0.7135 600e-6 579 end
  n         4 den=0.00125 1 595 end
  zirc2     5 1 579 end
  h2o       6 den=0.7135 1       579 end
  boron     6 den=0.7135 600e-6 579 end
  h2o       7 den=0.7135 1       579 end
  boron     7 den=0.7135 600e-6 579 end
  zirc2     8 1 579 end
  b4c       9 den=2.52 1 579 end
end comp
read celldata
  latticecell squarepitch pitch=1.4300 3
```

```
        fueld=0.9294  1                                                  28
        gapd=0.9484   4                                                  29
        cladd=1.0719  2 end                                              30
end celldata                                                             31
read sams                                                                32
nohtml                                                                   33
end sams                                                                 34
read model                                                               35
1/4 assembly model                                                       36
read parm                                                                37
  echo=yes saveangflx=yes                                                38
  epsinner= 1.00E-06                                                     39
  epsthrm=  1.00E-06                                                     40
  epsouter= 1.00E-06                                                     41
  epseigen= 1.00E-06                                                     42
  gptepsouter= 1.00D-06                                                  43
  prtxsec=         yes                                                   44
  sn=4                                                                   45
end parm                                                                 46
read materials                                                           47
  mix=1 pn=0 com="fuel"  end                                             48
  mix=2 pn=0 com="clad" end                                              49
  mix=3 pn=1 com="water" end                                             50
  mix=4 pn=0 com="gap" end                                               51
  mix=5 pn=0 com="guide tube" end                                        52
  mix=6 pn=0 com="CRout-clad" end                                        53
  mix=7 pn=0 com="CRout-abs" end                                         54
  mix=8 pn=0 com="CRin-clad" end                                         55
  mix=9 pn=0 com="CRin-abs" end                                          56
end materials                                                            57
read geom                                                                58
  unit 1                                                                 59
    com='fuel rod'                                                       60
    cylinder 10 .4647                                                    61
    cylinder 20 .4742                                                    62
    cylinder 30 .53595                                                   63
    cuboid   40  4p0.715                                                 64
    media 1 1 10                                                         65
    media 4 1 20 -10                                                     66
    media 2 1 30 -20                                                     67
```

```
  media 3 1 40 -30
boundary  40 2 2
unit 5
  com='guide tube'
  cylinder 10 .45
  cylinder 20 .52
  cylinder 30 .6502
  cylinder 40 .6934
  cuboid   50  4p0.715
  media 7 1 10
  media 6 1 20 -10
  media 3 1 30 -20
  media 5 1 40 -30
  media 3 1 50 -40
boundary  50 2 2
unit 11
  com='right half of fuel rod'
  cylinder 10 .4647   chord +x=0
  cylinder 20 .4742   chord +x=0
  cylinder 30 .53595  chord +x=0
  cuboid   40  0.715  0.0 2p0.715
  media 1 1 10
  media 4 1 20 -10
  media 2 1 30 -20
  media 3 1 40 -30
boundary  40 1 2
unit 12
  com='top half of fuel rod'
  cylinder 10 .4647   chord +y=0
  cylinder 20 .4742   chord +y=0
  cylinder 30 .53595  chord +y=0
  cuboid   40  2p0.715 0.715 0.0
  media 1 1 10
  media 4 1 20 -10
  media 2 1 30 -20
  media 3 1 40 -30
boundary  40 2 1
unit 51
  com='right half of guide tube'
  cylinder 10 .45   chord +x=0
```

```
    cylinder 20 .52    chord +x=0
    cylinder 30 .6502  chord +x=0
    cylinder 40 .6934  chord +x=0
    cuboid   50  0.715 0.0 2p0.715
    media 7 1 10
    media 6 1 20 -10
    media 3 1 30 -20
    media 5 1 40 -30
    media 3 1 50 -40
  boundary  50 1 2
  unit 52
    com='top half of guide tube'
    cylinder 10 .45    chord +y=0
    cylinder 20 .52    chord +y=0
    cylinder 30 .6502  chord +y=0
    cylinder 40 .6934  chord +y=0
    cuboid   50  2p0.715 0.715 0.0
    media 7 1 10
    media 6 1 20 -10
    media 3 1 30 -20
    media 5 1 40 -30
    media 3 1 50 -40
  boundary  50 2 1
  unit 53
    com='1/4 instrument tube'
    cylinder 10 .6502   chord +x=0 chord +y=0
    cylinder 20 .6934   chord +x=0 chord +y=0
    cuboid   40  0.715 0.0 0.715 0.0
    media 3 1 10
    media 5 1 20 -10
    media 3 1 40 -20
  boundary  40 1 1
  global unit 10
    com='1/4 assembly'
    cuboid 10 10.725 0.0 10.725 0.0
    array 1 10 place 1 1 0 0
    media 3 1 10
  boundary 10 15 15
end geom
read array
```

```
  ara=1 nux=8 nuy=8 typ=cuboidal pinpow=yes
  fill
    53  12 12 12 52 12 12 12
    11  1  1  1  1  1  1  1
    11  1  1  1  1  5  1  1
    11  1  1  5  1  1  1  1
    51  1  1  1  1  1  1  1
    11  1  5  1  1  5  1  1
    11  1  1  1  1  1  1  1
    11  1  1  1  1  1  1  1  end fill
end array
read bounds
  all=refl
end bounds
end model
end
=shell
cp ft42f001 ${RTNDIR}/ft42f001
cp ft92f001 ${RTNDIR}/ft92f001
cp senlib.sen ${RTNDIR}/senlib.sen
cp bonamist.sen ${RTNDIR}/bonamist.sen
cp worf ${RTNDIR}/worf
cp i_worker0002 ${RTNDIR}/i_worker0002
cp i_newt ${RTNDIR}/i_newt
end
```

Appendix–A/scale_inpt.inp