INL/EXT-16-37243

Light Water Reactor Sustainability Program

System Reliability Analysis Capability and Surrogate Model Application in RAVEN

Cristian Rabiti, Andrea Alfonsi, Dongli Huang, Frederick Gleicher, Bei Wang, Hany S. Abdel-Khalik, Valerio Pascucci, and Curtis L. Smith

November 2015

DOE Office of Nuclear Energy

DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

INL/EXT--16-37243

Light Water Reactor Sustainability Program

System Reliability Analysis Capability and Surrogate Model Application in RAVEN

Principal Investigator:
Cristian Rabiti (INL)

Topology-Inspired Batch Selection for Acceleration of Adaptive Sampling:
Bei Wang (University of Utah)
Valerio Pascucci (University of Utah)
Cristian Rabiti (INL)

***Theoretical Development and Implementation Results of Surrogate Construction Algorithms for RAVEN's Multi-Physics Models*:**
Dongli Huang (Purdue University)
Hany S. Abdel-Khalik (Purdue University)
Cristian Rabiti (INL)

Multi-Physics Surrogate Coupling:
Andrea Alfonsi (INL)
Cristian Rabiti (INL)

Project Overview and Coordination:
Cristian Rabiti (INL)
Curtis L. Smith (INL)

November 2015

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov/lwrs

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

EXECUTIVE SUMMARY

This report describes the effort performed to improve the analysis capabilities of the RAVEN code. These efforts include improving the reliability (or "limit") surface search of the RAVEN code and exploring new opportunities in usage of surrogate models by extending the current RAVEN capabilities to multi-physics surrogate models construction for high-dimensionality problems.

The RAVEN code has the capability of identifying reliability surfaces. In the text, we use the idea of limit or reliability surfaces Interchangeably. In general, a limit surface is a surface that separates the input space from where the system will evolve, either toward success or failure. Depending on how success or failure is defined, the limit surface could be a reliability surface. For example, when the success or failure corresponds to the availability or functionality of a system, the limit surface coincides with the reliability of the system. For a complex system (i.e., a nuclear power plant system), availability can be defined in many ways. If we account for the possibility of recovering the system over a certain amount of time, the reliability is an integral measure over time. For example, the definition of core damage can be taken as a reliability metrics due to the permanent impairment of the asset (i.e., the plant no longer functions due to the core damage).

An example of an input space is the recovery time of the auxiliary cooling system and the operating power level of a nuclear power plant. In a station blackout scenario, the combination of these two parameters, where we have a transition between core and no core damage, represents the limit/reliability surface. As we describe in this report, the RAVEN code is able to represent this surface through advanced simulation-based analysis.

In safety analysis, risk management and system design knowledge of the limit surface location provide an important guidance to engineers.

Unfortunately, determining the location of the reliability surface is a very complex problem that could easily become computationally untreatable. This complexity depends on the computational time needed to evaluate the goal function (i.e., the performance metric of interest) for a specific set of the input parameters (e.g., computing peak clad temperature for one RELAP-7 simulation of a loss-of-coolant-accident scenario), from the number of parameters in the input space, or from the probability associated with the failure region. To overcome these computational challenges, RAVEN has implemented several acceleration algorithms such as surrogate model-guided sampling of the input space, adaptive grid, and, finally, a sampling strategy from a topological analysis of the scoring function. The description of the last acceleration scheme is the first subject of this report. One major challenge in determination of the limit surface location is how to choose the sequence of points to be sampled. Current implementation has been improved by introducing a scheme where a scoring function is constructed to evaluate the most promising point to be explored and then a topological analysis of this function is performed to establish a batch of points that should bring the highest informational gain. This implementation has led to a faster and more robust reliability/limit surface-searching algorithm.

The report then focuses on two basic research activities that constitute the foundation of expanding the RAVEN capability to construct reduced-order models to a multi-physics problem and high-dimensionality problems for reliability analysis. We describe these capabilities, including situations where the current RAVEN approach for identification of the reliability surface might be problematic.

Section 3 of this report deals with the theory of surrogate model construction in cases where the complexity (number of degree of freedom used to describe the system) of the problem being analyzed is large. In particular, reactor physics (i.e., RattleSNake) and fuel performance (i.e., BISON) are chosen as two coupled physics. The developed theory is tested to analyze the possibility of successfully generating two separate surrogate models coupled via a very low-dimensionality space and, overall, still being able to reproduce the response of the original system.

The presented work is complete with respect to theoretical development; it has been successfully tested using BISON and RattleSnake. Full implementation of this capability within the RAVEN code has just started and will continue.

This work has importance in reliability analysis and also offers the possibility to deal with problems in uncertainty quantification and optimization.

Following an ensemble approach, an approach in RAVEN has been implemented and tested that allows construction of reduced order models of complex systems by assembling the separate reduced order models of each single component of the system. While RAVEN already has the capability to construct and store surrogate models for a given application, this feature has now been extended, allowing for multiple applications to be integrated and, thereby, representing a complex system without the need of the original codes to be coupled.

CONTENTS

EXECUTIVE SUMMARY v

ACRONYMS x

1. INTRODUCTION 1

2. ADAPTIVE SAMPLING IN RAVEN: TOPOLOGY-INSPIRED BATCH SELECTION 2
   - 2.1 Introduction 2
   - 2.2 Adaptive Sampling Pipeline for Reliability Surface Search in Serial and Batch Modes 3
   - 2.3 Designing Topology-Inspired Batch Selection 4
   - 2.4 Batch Selection Implementation in RAVEN 5
     - 2.4.1 Implementation of General Batch Selection 6
     - 2.4.2 Implementation of Topology-inspired Batch Selection 7
   - 2.5 Reliability Surface Thickening and Additional Features 7
   - 2.6 Testing 8
     - 2.6.1 Testing Environment 8
     - 2.6.2 Experimental Observations 10
   - 2.7 Discussion 15

3. THEORETICAL DEVELOPMENT AND IMPLEMENTATION RESULTS OF SURROGATE CONSTRUCTION ALGORITHMS FOR RAVEN'S MULTI-PHYSICS MODELS 15
   - 3.1 Introduction 15
   - 3.2 Background 16
   - 3.3 Range Finding Algorithms for Multi-Physics Models 23
   - 3.4 Surrogate Model Construction 26
   - 3.5 Numerical Results 27

4. META-MODEL IMPLEMENTATION IN RAVEN FRAMEWORK 37
   - 4.1 Introduction 37
   - 4.2 Models in RAVEN 37
   - 4.3 Meta-Model in RAVEN 38
     - 4.3.1 Implementation 40
     - 4.3.2 Meta-model Resolving in a Non-Linear System 40
   - 4.4 Application Example 41
   - 4.5 Final Remarks 43

5. CONCLUSION 43

6. REFERENCES 44

FIGURES

Figure 1. Example of multi-physics coupling with a high density field exchange. 1

Figure 2. AS pipelines for RS search: serial mode (left) and batch mode (right). Training points are marked by crosses, RS candidates are marked by circles with selected ones marked in red. 4

Figure 3. Left: topology-inspired batch selection AS pipeline for RS search. Right: (a) naive strategy selects the top b candidates with the highest scores; (b) local maxima and local minima paired by persistence, e.g., the red points have persistence valued at h – their height difference; (c) maxV strategy selects the top b local maxima with the highest scores; and (d) maxP strategy selects the top b local maxima with the highest persistence values. 5

Figure 4. Parallel implementation of AS in RAVEN, where the process (marked within a grey box) of querying the oracle with a selected candidate is typically a time-consuming step within the pipeline. 6

Figure 5. Using batch buffer to simulate batch selection in RAVEN. 6

Figure 6. Using batch buffer to simulate topology-inspired batch selection in RAVEN. Modification with respect to the general batch selection is highlighted in yellow. 7

Figure 7. Thickening of RS for exploration. On the left: the first and second layers of RS candidates on a simple grid are highlighted with dark and light colors respectively. On the middle and on the right: an RS is visualized with a thickening parameter valued at 1 and 2 respectively. Training points are represented by crosses or triangles, adaptively sampled points are black circles, and RS candidates are solid circles colored by their scores (high scores are in red while low scores are in blue). 7

Figure 8. Visualization of the distance scoring function over the entire domain during the four initial training stages for the Circle dataset (using a batch size of 4). Red corresponds to high and blue corresponds to low function values. Regions with the lowest function values overlap with locations of the training points. 8

Figure 9. Testing datasets: 2D functions with relatively simple RS boundaries. 8

Figure 10. Testing datasets: 2D functions with complex RS boundaries. 9

Figure 11. Visualizations of the data domain during several early stages of the AS process for the Face dataset using 4 maxP strategy. We visualize the locations of the current training points (in triangles), selected RS candidates (in black circles, also pointed by arrows) and the reliability surface (red means high and blue means low function values). During each iteration, we choose the top 4 local maxima with the highest persistence. The selected RS candidates are shown to be well separated and sometimes belong to different connected components of the RS. 10

Figure 12. Convergence plot of the Face dataset using the weighted distance scoring with 10 processors, with various zoomed- in views. X-axis indicates the number of points in the training set; Y-axis shows the F1-score. Each colored curve corresponds to the convergence behavior of an AS strategy. The thick bars pointed by the black arrow in (b) reflect when the candidates are obtained from a MC (instead of an adaptive) sampling procedure. 11

Figure 13. Convergence plots of the Face dataset using the weighted distance scoring with 10 processors with zoomed-in views. 12

Figure 14. Convergence plots of the Hubble dataset using the weighted distance scoring with 10 processors. 13

Figure 15. Convergence plots of the Circle and GMM datasets using the weighted distance scoring with 10 processors. 14
Figure 16: Active Subspace Reconstruction Error 22
Figure 17. Multi-Physics MAMMOTH-based Model of RattleSnake and BISON. 23
Figure 18. Numerical meshes employed by RattleSnake and BISON. 27
Figure 19. RattleSnake-BISON output distributions. 28
Figure 20. Maximum relative reduction error vs. size of active subspace for Scenarios 1 (b), 2 (c), and for the cross section active subspace (a). 31
Figure 21. Maximum relative reduction error vs. Size of active subspace for fission rate and burnup (on the left Case 1, on the right Case 2). 32
Figure 22. Maximum relative reduction error vs. size of active subspace for temperature (on the left Case 1, on the right Case 2). 33
Figure 23. Maximum relative reduction error vs. size of active subspace (Case 3). 34
Figure 24. Relationship between the DoFs of RattleSnake and BISON. 36
Figure 25. First two DoFs of power density. 36
Figure 26. Example of a meta-model constituted by three sequential sub-models. 39
Figure 27. Example of a meta-model constituted by three sub-models, two of which can be run independently. 39
Figure 28. Meta-model data exchange among sub-models. 40
Figure 29. Meta-model resolving in a non-linear system of equations – Picard's iterations. 41
Figure 30. Heat conduction meta-model connections in a chain of evaluations. 42
Figure 31. Heat conduction meta-model connections in a non-linear system. 42
Figure 32. Temperature at mid-plane: sequential model (left) and Picard iteration (right). 42
Figure 33. Compute thermal conductivity: sequential model (left) and Picard iteration (right). 43

TABLES

Table 1. Initial fuel pin parameters. 27
Table 2. Reduction scenarios. 29

ACRONYMS

AS	adaptive sampling
BB	batch buffer
DoF	degrees of freedom
HPL	hanging point list
MC	Monte Carlo
RFA	range finding algorithm
RS	reliability surface
ROM	reduced order model

System Reliability Analysis Capability and Surrogate Model Application in RAVEN

1. INTRODUCTION

This report illustrates the details of three major theoretical developments in support of the RAVEN project and, in particular, the reliability surface analysis capability. The first development is focused on improvement of the research speed of the limit (i.e., reliability) surface location. Previous reports, as for example 10, discuss how the limit surface location reveals crucial information to engineers who perform risk management. Speed and the accuracy of the limit surface location are crucial components of RAVEN functionality. In this report, a few strategies have explored improvements of the performance of searching algorithms to take advantage of parallel machines and to introduce a new approach in selection of the next point to be sampled. Theoretical background, new calculation flows, and results for theoretical examples are described in the first section of this report. The second and the third developments presented in this report are related to and are part of an overall long-term development plan for RAVEN, where reduced order models (ROMs) could be used dynamically in analyzing complex multi-physics system reliability.

Figure 1 represents a system where two physics are coupled to generate the overall representation of the global system. Each physic is represented by a separate model and the information is exchanged between the two models through high-density fields containing millions of degrees of freedom (DoF). As high density field here we identify a scalar or vectorial field described by a large number of points/degrees of freedom (up to several billions).

In the approach followed by RAVEN, the overall system would be examined and, possibly, the relationship between the input space and the engineering figure of merits is emulated by ROMs to speed up processes such as reliability surface search, optimization, model tuning, and uncertainty propagation. This approach is not always feasible or practical. The reason for this is the more the complexity of the system grows (i.e., number of internal DoF), the higher the computational time that is required to examine the response of the system for a realization of the input space parameters. Generally, the computational effort grows more than linearly; therefore, if possible, it would be more effective to create a ROM separately for each model rather than the full system and then perform coupling later on. Other cases where it could be convenient to use this approach are when the codes implementing the single models are not directly coupled. In order to implement such a scheme, several expansions to current RAVEN capabilities and theoretical developments are needed.

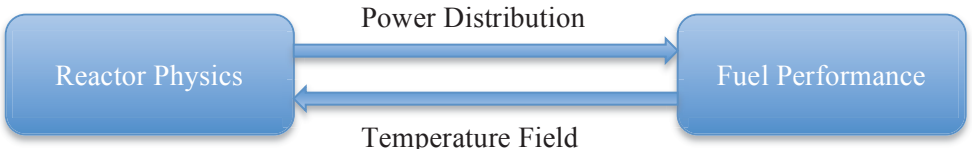


Figure 1. Example of multi-physics coupling with a high density field exchange.

One major theoretical challenge to be solved is how to construct ROMs to emulate models producing high-density fields with millions, or more, of DoF instead of having to represent the response of the system via a few figures of merit. The proposed approach is to exploit the fact that a high number of DoF usually hides a high level of correlation among single DoF. This is because every physical system presents a natural tendency to diffuse the information; therefore, the true number of DoF is much less than the number it is used to represent in the response of the model. To detect the structure of those correlations and to detect the minimal set of DoF that allows us to characterize the model response, this work proposes an approach based on analysis of the covariance of several high-density fields produced by perturbation of the input space. This work proves the feasibility of the approach and will be followed by implementation of such capabilities in RAVEN.

Another situation where reliability analysis would benefit from construction of a complex system representing assembly of separate surrogate models is when the coupling between the high-fidelity representations of the system components is not available. In this case, later advancements in the RAVEN infrastructure that are presented here offer a solution.

Until now, in RAVEN, it has been possible to create surrogate models of different models with several input parameters and output (i.e., figures of merit) and store them for later usage. Now it is possible to combine them to create representations of complex systems that were not available before.

This work currently covers only sequential and non-linear steady-state coupling; however, once the time-dependent surrogate models are completed in RAVEN, the work could be extended to time-dependent analysis and even high-density, field-based coupling.

In order to plan the deployment of such theoretical work inside RAVEN, it is necessary to begin implementation of the needed software infrastructure. Currently, a new infrastructure has been put in place in the RAVEN framework, which allows coupling of several different surrogate models, laying down the skeleton for future implementation of multi-physics coupling for high-dimensionality fields.

2. ADAPTIVE SAMPLING IN RAVEN: TOPOLOGY-INSPIRED BATCH SELECTION

2.1 Introduction

Nuclear simulations typically contain a large number of uncertain parameters and can be computationally expensive. During simulation-based risk analysis and uncertainty quantification, it is important to understand the relationship between the input and output of a simulation using as few simulations as possible. This is a typical context for adaptive sampling (AS), where active selection of training points can significantly reduce computational effort and accelerate the learning process of the response surface of a given simulation.

During such a learning process:

- Few observations (runs of the high fidelity model) are obtained as the initial training set
- ROM is constructed (trained using the available observations) to represent the simulation space either as a regressor or a classifier
- Label (output) a set of unlabeled points (points of the input space for which the corresponding output is unknown) is predicted using the ROM
- Unlabeled points are ranked and selected based on some scoring/utility function with respect to the value as next training points
- Evaluation of the selected point is added to the training set
- ROM is retrained with the updated training set
- Process is repeated until convergence is achieved (stability of the ROM).

Therefore, we attempt to gain the most information with a small number of carefully selected sampled points, limiting the number of expensive queries from the simulation. In this report, we focus specifically on identifying the reliability surface, that is, the boundaries in the simulation space between system failure and system success. More in detail, the focus is in constructing an ROM of type classifier capable to predict the location of the reliability surface with a very low number of samples of the high fidelity model.

In particular the following improvements are suggested:

- Expansion of the current multi-processor AS capabilities to include a general batch selection framework
- Introduction of a collection of topology-inspired batch selection AS techniques that could be adapted by any scoring function
- Enabling of reliability surface (RS) thickening to allow adjustment between exploration and exploitation
- Extra visualization capabilities that visualize the locations of training points, adaptively sampled points, reliability surfaces, and the landscape of scoring functions during the AS process of 2D datasets for testing and debugging purposes.

The proposed techniques are considered exploratory and potentially complementary to some of the existing AS capabilities in RAVEN.

In the following sections, we review the AS pipeline for RS search and introduce the batch mode selection in Section 2.2. The topology of the scoring function is discussed as a class of topology-inspired batch selection strategies in Section 2.3. In Section 2.4, we focus on implementation details of general and topology-inspired batch selection in RAVEN. RS thickening is described in Section 2.5, together with additional visual capabilities for testing and debugging. We experiment with a set of testing datasets and summarize the strengths and weaknesses of our proposed approaches in Section 2.6, and we discuss future research directions in Section 2.7.

2.2 Adaptive Sampling Pipeline for Reliability Surface Search in Serial and Batch Modes

In a typical setting for RS search, new training samples are selected in serial, as illustrated in Figure 2 (left). We begin by selecting a limited number of initial training points (via forward sampling strategies such as Monte Carlo) and obtain their labels by querying the high-fidelity simulation code (e.g., the computationally expensive RELAP-7), referred to as the *labeling source* or the *oracle*. Second, a ROM classifier (e.g., Support Vector Machines) is trained with the given training set. Third, the ROM is used to predict the labels for all grid points in the domain space and a set of candidate points surrounding the RS is identified based on these predicted values. Finally, each RS candidate is ranked based on an AS scoring function (usually derived from qualitative or quantitative relations between the training points, and their observed and predicted values), and a single candidate is selected to be added to the training set to begin a new round of iterative learning process. On the other hand, sometimes for parallel labeling environment or models with slow training procedures, batch mode AS allows us to query instances in groups.[1] As illustrated in Figure 2 (right), instead of selecting candidate one at a time, a batch size of b candidates are selected at the same time to be added to the training set during the iterative process.

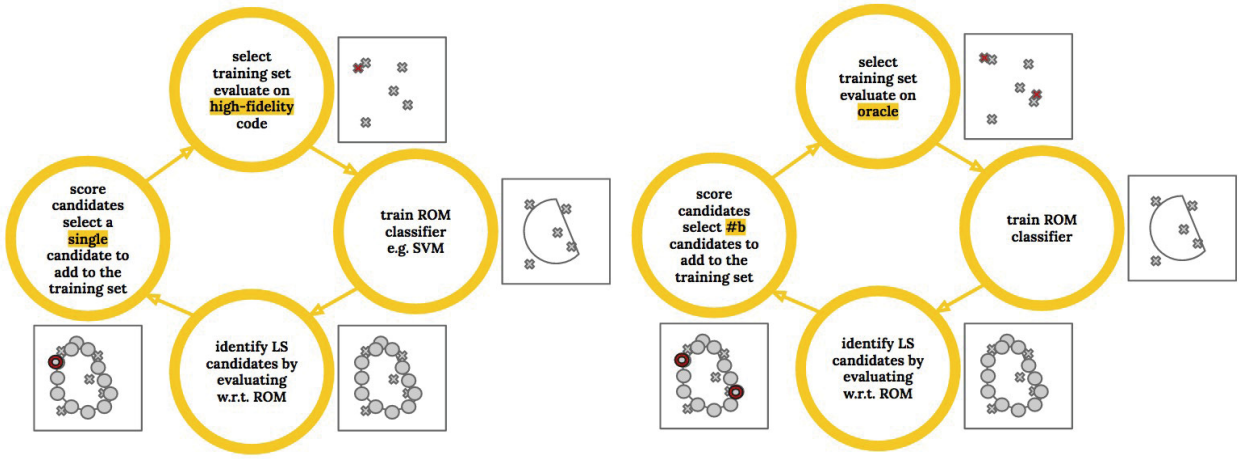


Figure 2. AS pipelines for RS search: serial mode (left) and batch mode (right). Training points are marked by crosses, RS candidates are marked by circles with selected ones marked in red.

Batch mode AS may be more efficient than the serial mode under certain scenarios from a computational point of view. Sometimes retraining a classifier whenever a new training point is added can be time-consuming, for example, with large ensemble models and complex models for structured prediction tasks,[1] it may be more efficient to select and label a set of points before the retraining process.[2] In addition, scoring functions (e.g., based on error and variance reduction) may be prohibitively expensive to evaluate.[1] Further, under some experimental procedures, multiple labels can become available simultaneously; and batch mode takes advantage of such a parallel labeling instance.[2] Essentially, at the moment when the training of the surrogate model and the evaluation of the scoring cost have a computational cost not negligible with respect the evaluation of the oracle a batch approach is preferred.

In this report, we explore the general batch selection as well as topology- inspired batch selection processes in RAVEN, to understand their strengths and weaknesses.

2.3 Designing Topology-Inspired Batch Selection

The main challenge in batch mode AS is how to carefully select a set of batched candidates. For a fixed batch size b > 1, selecting the best b candidates based on their scores, referred to as the naive strategy, usually does not work well because it does not consider the overlap in information content among the "best" instances.[1] Existing algorithms use diversity,[2] density,[3] or gradient search[4] in their batch constructions. In our exploration, we take into consideration the topology of the scoring function in our batch construction where points are selected from regions with distinct topological features.

The topology-inspired batch selection pipeline is illustrated in Figure 3 (left). It is just a minor modification from the general batch selection process in Figure 2 (right) where the choice of candidates relies on some topological feature of the scoring function. Take a one-dimensional scoring function for example, as the one shown in Figure 3 (right), the naive strategy in (a) may choose two candidates that are virtually identical with almost redundant information; while topology-inspired strategies in (c) and (d) construct batches from topologically distinct regions. In (c), candidates are chosen among the local maxima of the scoring function with the highest function values, referred to as the maxV strategy; while in (d) they are chosen to be the top b local maxima with the highest persistence values, referred to as the maxP strategy. Here, persistence[5,6,7] is a topological notion that quantifies the significance of topological features of a function; local maxima with high or low persistence values are considered topological signals or noise respectively within their local neighborhoods. Yellow points with the highest persistence values in (d) therefore represent locally distinct and well-separated regions within the "landscape" of the

given scoring function; therefore selecting these points has the potential to maximize information gain. For a brief introduction to the notion of persistence, see Reference 8.

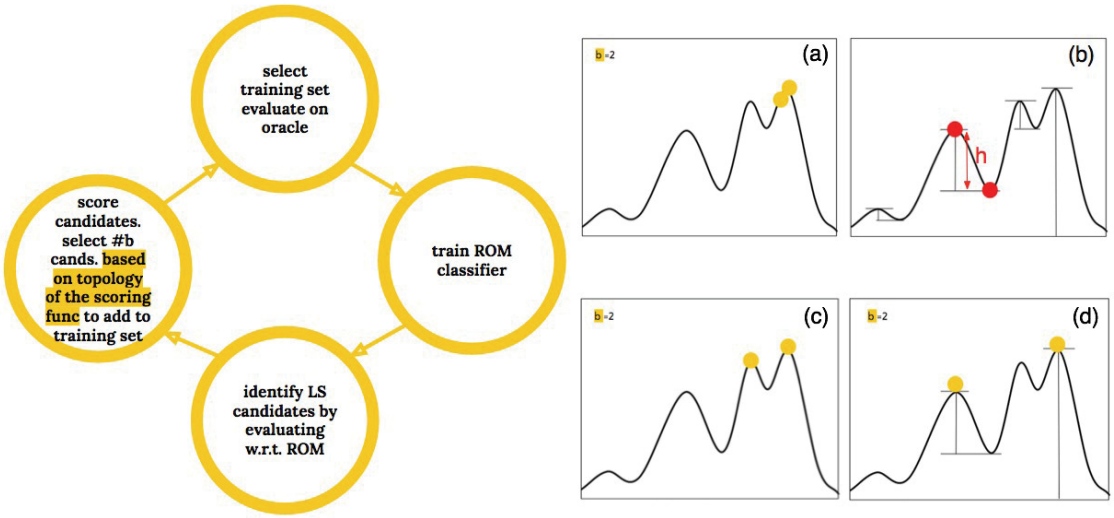


Figure 3. Left: topology-inspired batch selection AS pipeline for RS search. Right: (a) naive strategy selects the top b candidates with the highest scores; (b) local maxima and local minima paired by persistence, e.g., the red points have persistence valued at h – their height difference; (c) maxV strategy selects the top b local maxima with the highest scores; and (d) maxP strategy selects the top b local maxima with the highest persistence values.

Our framework is independent from and could be adapted for any scoring function as long as they allow for topological representation. The strategies mentioned here are just simple examples of a large class of topology-inspired ones that may be suitable for the batch mode selection. In addition, since the detection of local maxima and computation/approximation of persistence generalizes to high dimensions,[9] these strategies are applicable for high dimensional AS as well.

2.4 Batch Selection Implementation in RAVEN

We rely on the existing parallel (multi-processor) AS framework (as described in Reference 10) to simulate the general batch selection as well as the topology-inspired batch selection in RAVEN. The parallel framework in RAVEN relies on a specific data structure called a hanging point list (HPL) that keeps track of candidate points submitted for evaluation. In a typical scenario, multiple processors are available as computational resources during the AS process (e.g., we use 10 processors in our experiments). As illustrated in Figure 4, all processors are available at the beginning of the AS process. Suppose Processor 1 is the first processor that is available, it follows the serial AS pipeline from left to right, trains the ROM with the current set of training points, identifies RS candidate points, then scores the candidates and selects the top one candidate x that is not in the HPL. It pushes x to the HPL and query the oracle with x. After the query finishes its execution and an observation (i.e., true response) is obtained for x from the oracle, removes x from the HPL, and finally adds x to the training set. While Processor 1 is at some stage of its AS pipeline, due to the parallel implementation, Processor 2 (a second available processor), is running its own AS pipeline independently. Since querying the oracle with a selected candidate x is typically time- consuming, Processor 1 and 2 (and potentially additional processors) could be running simultaneously (Figure 4). Processor 2 therefore relies on the HPL to keep track of RS candidates already submitted for evaluation to avoid evaluating the same point twice. Such a parallel implementation using HPL is quite efficient in the AS process, since a given processor always takes advantage of the best available information (i.e., an updated training set) during its ROM training stage

and greedily selects the top candidate for evaluation by the oracle. This is referred to as the greedy strategy.

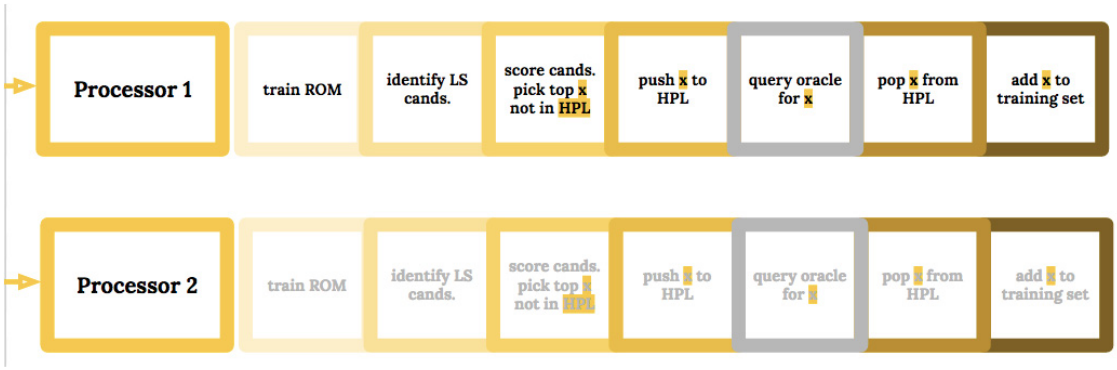


Figure 4. Parallel implementation of AS in RAVEN, where the process (marked within a grey box) of querying the oracle with a selected candidate is typically a time-consuming step within the pipeline.

2.4.1 Implementation of General Batch Selection

Given the above AS parallel framework, we now focus on simulating a general batch selection in RAVEN with a data structure called a batch buffer (BB). The batch selection size b is therefore controlled by the size of the BB. As illustrated in Figure 5, when a processor (e.g., Processor 1) becomes available, it first checks to see if the BB is empty. If the BB is empty, the processor proceeds by following the serial AS pipeline (Figure 4 top) with minor modifications. That is, after RS candidates are identified, the processor selects the top b candidates (based on a given ranking criteria) and adds them to the BB; it takes a candidate x from the BB, pushing x to the HPL, obtaining its evaluation from the oracle, then removing x from the HPL and adding it to the training set. On the other hand, if the BB is not empty, then the processor proceeds by taking a candidate x from the BB and starts the evaluation process without retraining the ROM. Each of the other available processors follows the same pipeline in parallel, where the ROM is only retrained when the BB becomes empty. The addition of the BB to the existing parallel implementation therefore introduces a delayed effect within the first few stages of the AS pipeline, namely, retraining the ROM, identifying and scoring the RS candidates. At a first glance, using the BB for the batch mode selection does not appear to have obvious advantage over the existing framework; however, as demonstrated in Section 2.6, under certain scenarios, a batch mode selection with carefully selected candidates could potentially outperform the greedy strategy.

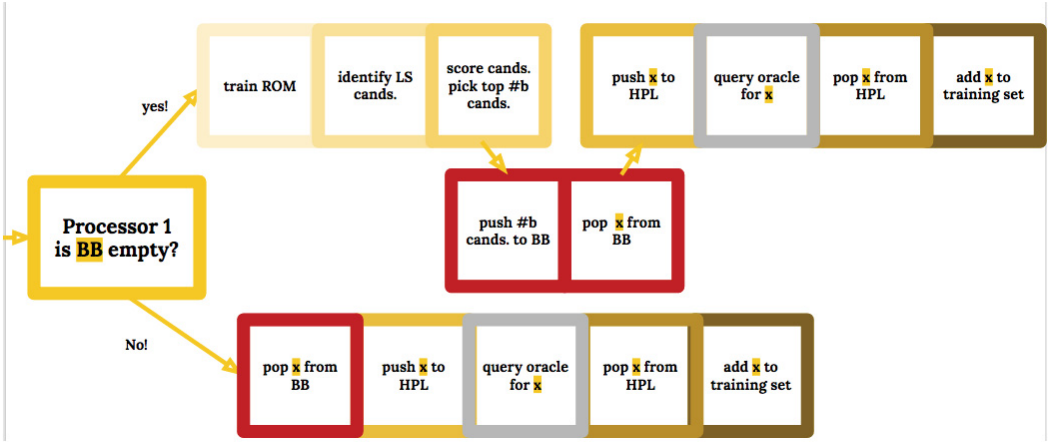


Figure 5. Using batch buffer to simulate batch selection in RAVEN.

2.4.2 Implementation of Topology-inspired Batch Selection

Topology-inspired batch selection in RAVEN adapts the general batch selection framework and takes the topology of a given scoring function into consideration. Its implementation details are illustrated in Figure 6, by making small modifications and incorporating topology-inspired batch construction within the general framework of Figure 5. To detect local maxima among the RS candidates and to compute their persistence, we employ the underlying grid as the combinatorial structure that connects the candidate points, and use the established Morse-Smale approximations for our computations. In a nutshell, a point whose scoring function value is higher than all of its grid neighbors are considered a local maximum; and its persistence is approximated by the function value difference between itself and its nearby saddle point (see Reference 9 for details).

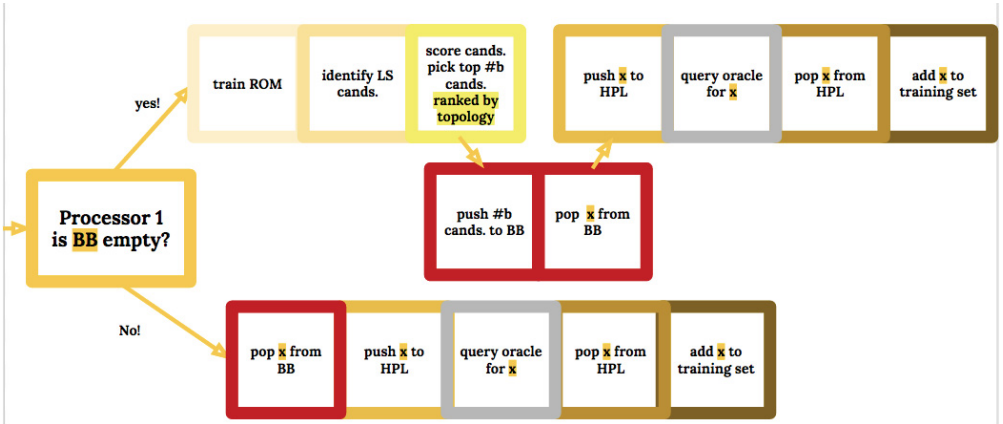


Figure 6. Using batch buffer to simulate topology-inspired batch selection in RAVEN. Modification with respect to the general batch selection is highlighted in yellow.

2.5 Reliability Surface Thickening and Additional Features

In the existing AS implementation,[10] the location of the RS is not exactly determined but rather bounded by a layer of grid points within its proximity, which in turn forms a set of RS candidates. By introducing a thickening parameter, we enlarge the search space of the RS into multiple layers of candidates, therefore allowing adjustment between exploration and exploitation (see Figure 7 for an example).

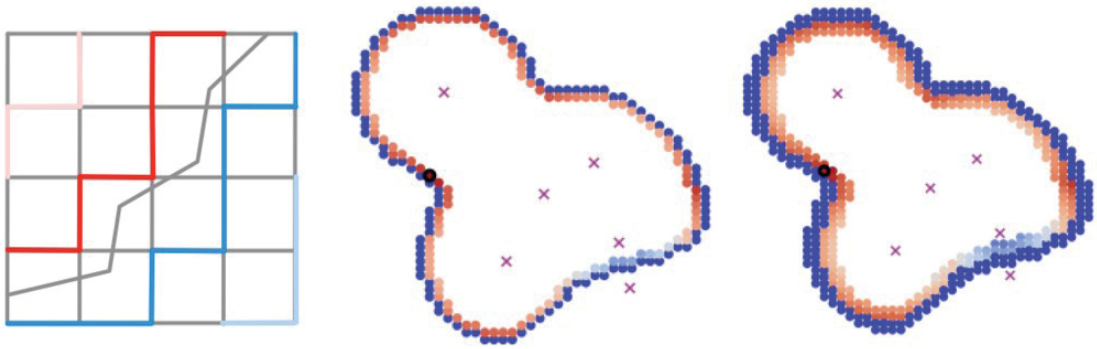


Figure 7. Thickening of RS for exploration. On the left: the first and second layers of RS candidates on a simple grid are highlighted with dark and light colors respectively. On the middle and on the right: an RS is visualized with a thickening parameter valued at 1 and 2 respectively. Training points are represented by crosses or triangles, adaptively sampled points are black circles, and RS candidates are solid circles colored by their scores (high scores are in red while low scores are in blue).

We also add extra visualization capabilities in RAVEN during the AS process of two-dimensional datasets for testing and debugging. Take Figure 7 middle for an example, for a single AS iteration, we mark the locations of training points, adaptively sampled points and reliability surfaces; as well as visualize the scoring functions restricted to the RS and the entire domain (Figure 8).

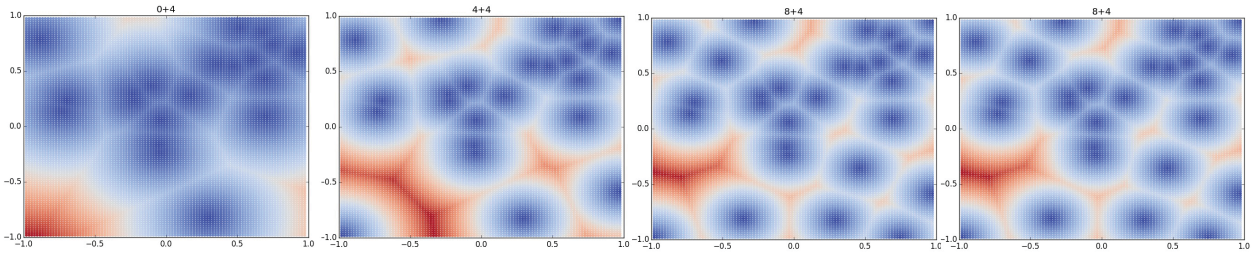


Figure 8. Visualization of the distance scoring function over the entire domain during the four initial training stages for the Circle dataset (using a batch size of 4). Red corresponds to high and blue corresponds to low function values. Regions with the lowest function values overlap with locations of the training points.

2.6 Testing

2.6.1 Testing Environment

2.6.1.1 Testing Datasets. We provide a collection of 2D testing datasets for evaluating the strengths and weaknesses of our proposed approaches. We test on 2D functions with simple (Figure 9) and complex (Figure 10) RS boundaries. These functions serve as the oracles within our AS pipeline and they are queried to obtain the true responses from selected candidate points. The Circle, GMM (the name is chosen since the function is generated as a Gaussian Model Mixture), and Salomon datasets are generated from functions with closed forms. The RELAP-7 and Islands datasets are inherited from the set of existing AS testing cases in RAVEN, where points are uniformly sampled in the cumulative distribution function range space, not in the domain space. The Hubble and Face datasets originate from grey-scale images. RSs (colored in purple) are defined by certain function value thresholds that create interesting boundaries, in particular, a large number of "islands" (connected components) exist for the Hubble and Face datasets, increasing the topological complexity of their corresponding reliability surfaces.

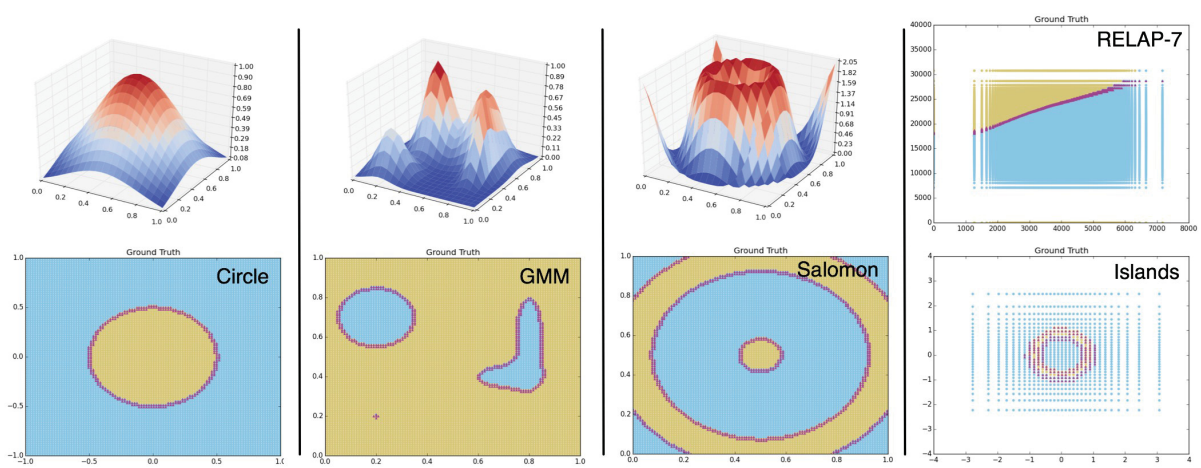


Figure 9. Testing datasets: 2D functions with relatively simple RS boundaries.

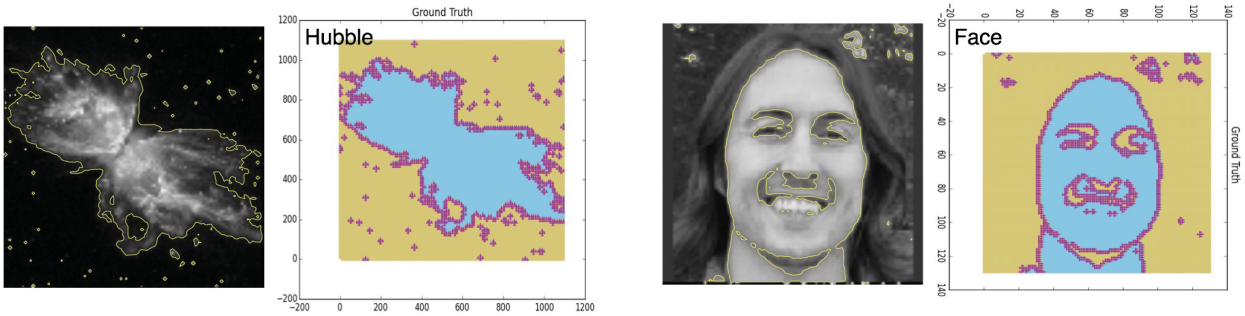


Figure 10. Testing datasets: 2D functions with complex RS boundaries.

2.6.1.2 Scoring Functions. Our proposed topology-inspired batch selection is independent of the choice of a scoring function. Various scoring/utility functions exist in literature[1] that measure uncertainty or information content associated with a given candidate. For example, in RAVEN, the score of an RS candidate point is based on its distance to the nearest training point (that has been or is being explored [i.e., points in the training set and the HPL]) weighted by the number of times its predicted label has changed during the iteration – such a strategy is referred to as the weighted distance scoring. Its simpler unweighted version, that is, the one based on just the distance between an RS candidate and its nearest training point, is referred to as the distance scoring. The weighted distance scoring is a fine-tuned scoring function that works particularly well for the RS search/classification problem. It is similar to the straddle scoring function[11,12] (and its variation in Reference 13) that is specially designed to target the RS recovery problem, which scores points based on a mixture of high predicted variance and a predicted mean near the threshold value. Due to the parallel implementation of AS in RAVEN using HPL, the greedy strategy using the weighted distance scoring function bears some resemblance to the believer strategy[13,14] that we have explored in some previous work,[15] since the points in the HPL (the ones being evaluated by the oracle) are factored into the computation of the scoring function.

2.6.1.3 Convergence Criteria and Monte Carlo Sampling. In the AS framework of RAVEN, an iterative procedure achieves its convergence when there are no changes in the location of the RS after a certain number (e.g., 25) of consecutive iterations.[10] To measure the final accuracy of our extracted RS at the time of convergence, we use the F1-score,[16] which is a function of precision and recall. Let TP, FP, FN denote true positives, false positives and false negatives in comparing the extracted RS from the ROM and the ground truth. The precision p equals TP/(TP + FP), while the recall r is defined by TP/(TP + FN). The F1-score is then defined by 2pr/(p + r). In RAVEN, Monte Carlo (MC) sampling can be used to generate an initial set of training points until an ROM (e.g., Support Vector Machine classifier) could be properly trained. In addition, MC sampling kicks in whenever the RS candidate set does not have any "new" samples, that is, whenever all of the candidates have zero scores (because they are either in the training set or in the HPL, or they have never changed labels in the training process). Such MC sampling typically occurs toward the end of the AS process when it is near convergence to ensure that the region of interest has been sufficiently explored (such a process may have interesting consequence as we point out later).

2.6.1.4 Adaptive Sampling Strategies. For a given dataset, we experiment with a variety of AS strategies (described in Section 4). This includes the greedy strategy, the topology-inspired batch selection strategies with batch sizes of 4 and 8, and strategies involving the thickening of the RS with 2 layers. For example, 4_maxv is the maxV strategy with batch size 4, tg corresponds to the greedy strategy with a thickened RS, and t8_maxp is the maxP strategy with a thickened RS and batch size 8. As mentioned in Section 3, naive batch selection which chooses the b-best instances typically does not work well due to information overlap; as a sanity check, we still include the naive batch strategy with batch sizes 4 and 8 in a few of our testing cases for comparison purposes. For a single AS trial, to obtain

consistent results for comparison, we use the same and sufficiently large (e.g., 20) initial training set (such that the ROM is properly trained from the beginning) across all AS strategies.

2.6.1.5 Other Experimental Details. For a given dataset, we run a number of AS trials (e.g., 5 or 10) until convergence with randomly generated initial training set. We obtain an F1-score convergence plot for each trial, whose x-axis corresponds to the number of points in the training set and y-axis is the F1-score of the RS at the end of each (batched) iteration. We also compute a point-wise median F1-score plot across all trials. However, it turns out that such a median plot introduces hard-to-interpret biases and is not as informative as what we would expect. Since MC sampling influences how we interpret our results, we add additional visualization (i.e., horizontal bars colored by the corresponding AS strategy) in the plot when MC sampling takes place during an AS iteration.

2.6.2 Experimental Observations

The default experimental setting involves multiple processors (e.g., 10) using the weighted distance function. To understand the effect of the existing parallel implementation on the AS process, we experiment with our testing datasets under the setting of a single processor and multiple processors respectively. Since our topology-inspired strategies do not rely on specific choices of scoring functions, we also experiment with both the distance scoring and the weighted distance scoring for performance comparisons. We summarize our key observations below.

The main conclusion is that the greedy strategy typically performs very well for a simple RS, while the topology-inspired batch selection strategies could potentially outperform the greedy strategies for a complex RS.

2.6.2.1 For Complex RS, Topology-Inspired Batch Selection Can Sometimes Outperform the Greedy Strategy. For an RS that contains a number of "islands" (connected components) or has boundaries with complex geometry, for example, in the case of the Face and Hubble datasets, topology-inspired batch selection can outperform the default, greedy strategy, in particular, within the "intermediate" stages of the AS process. The performance gain could be largely contributed to the topology-inspired batch selection being able to explore the domain more efficiently by selecting multiple candidates from well-separated (and in some cases, independent) regions (see Figure 11 for an example). On the other hand, although the greedy strategy always takes advantage of the updated training set for the ROM at the beginning of each iteration, the fact that it only chooses the top one candidate sometimes limits its exploratory ability.

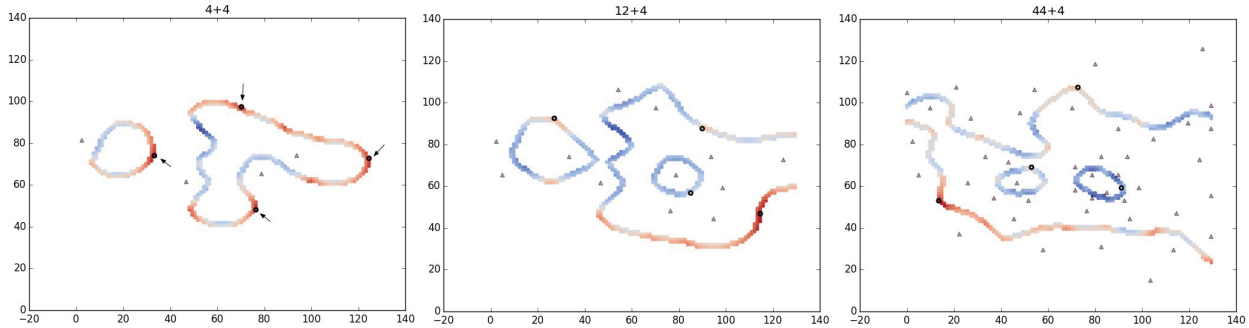


Figure 11. Visualizations of the data domain during several early stages of the AS process for the Face dataset using 4 maxP strategy. We visualize the locations of the current training points (in triangles), selected RS candidates (in black circles, also pointed by arrows) and the reliability surface (red means high and blue means low function values). During each iteration, we choose the top 4 local maxima with the highest persistence. The selected RS candidates are shown to be well separated and sometimes belong to different connected components of the RS.

Take Figure 12 for an example using a particular trial of the Face dataset. To explain our convergence results better, we divide the x-axis of its convergence plot into three (data-dependent) regions: the early convergence region describes convergence behavior when the number of training points is between 0 and 200; the intermediate convergence region contains roughly 250–1000 training points; and the end convergence region describes convergence behavior beyond 1000 training points. As illustrated in Figure 12(c), the topology-inspired techniques, namely, t8_maxP (dark purple), 8_maxP (light purple), and 4_maxV (light orange) all outperform the greedy strategy (light blue) within the intermediate convergence region. The greedy strategy is only able to have drastic performance improvement with the help of some MC samplings (as visualized by the light blue bars in Figure 12b) within the end convergence region. In other words, the MC strategy helps the greedy strategy to get "unstuck" from locally optimal but globally unsatisfactory scenarios (as detailed in Section 2.6.2.3). In addition, in this example the naive strategies, Naive 4 (light green) and Naive 8 (dark green) do not perform very well due to the information overlap as discussed in Section 2.3. However, it is hard to compare the performances among the various AS strategies within the early convergence region (Figure 12d), as they all appear to perform comparably with wide fluctuations followed by stable convergences. We have some evidence that topology-inspired batch selection strategies help to explore a large region of the domain space during the early convergence stage (e.g., Figure 11); however systematic study is left for the future work.

For completeness, we also include two other trials for the Face dataset under the same setting with randomized initial training sets in Figure 13. In Figure 13(a), the topology-inspired 4_maxV (light orange) and 8_maxV (yellow) outperforms the greedy strategy (light blue) within the intermediate convergence region. The greedy strategy is only able to improve its performance by relying on a MC sampling toward the end convergence region. It is also interesting to note that in this particular trial, the naïve_8 strategy (i.e., naive batch selection with batch Size 8, dark green) also outperforms the greedy strategy for a significant period of time within the intermediate region.

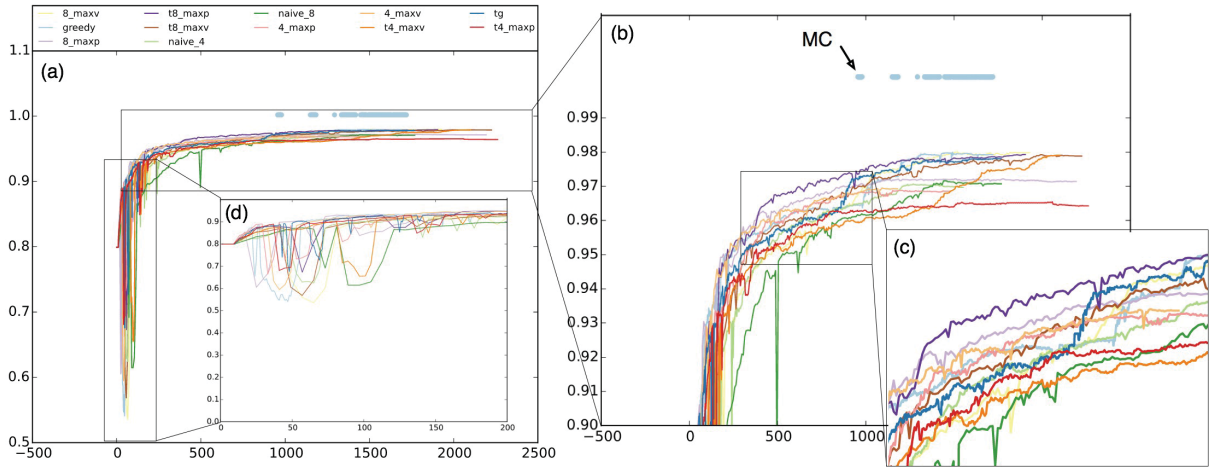


Figure 12. Convergence plot of the Face dataset using the weighted distance scoring with 10 processors, with various zoomed- in views. X-axis indicates the number of points in the training set; Y-axis shows the F1-score. Each colored curve corresponds to the convergence behavior of an AS strategy. The thick bars pointed by the black arrow in (b) reflect when the candidates are obtained from a MC (instead of an adaptive) sampling procedure.

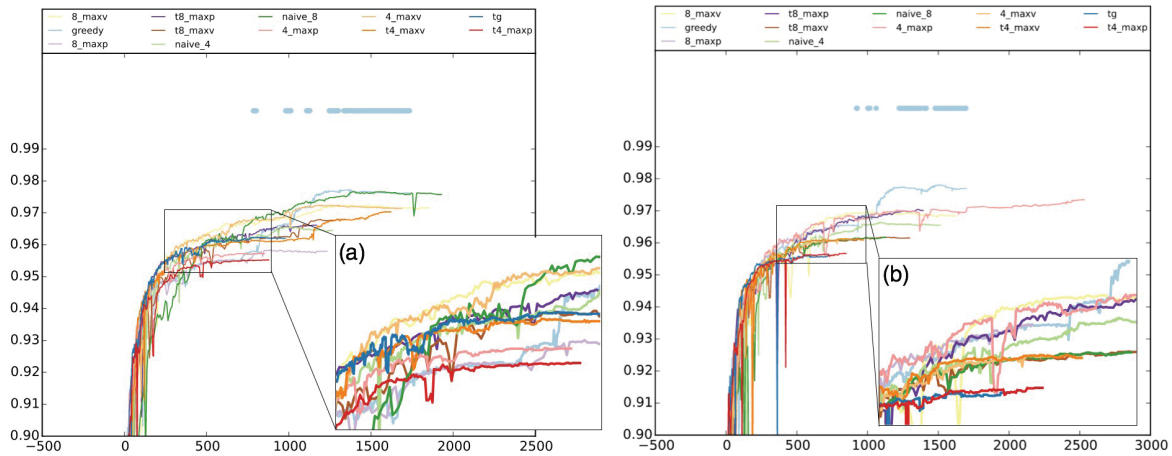


Figure 13. Convergence plots of the Face dataset using the weighted distance scoring with 10 processors with zoomed-in views.

This is possibly due to the fact that the naive batch selection with a large enough batch size could sometimes get "lucky" in exploring unknown regions. In Figure 13(b), a couple of topology-inspired strategies, noticeably 4_ maxP (pink) and 8_maxV (yellow) outperform the greedy strategy within the intermediate convergence region until the MC sampling kicks in to help improve the performance of the greedy strategy.

We also showcase several trials of the Hubble dataset (Figure 14). Figure 14 (a), (b), and (d) give some evidence that the topology-inspired batch selection strategies could sometimes outperform the greedy strategy (light blue): (a) and (d) demonstrate that the visible performance gains are within the intermediate convergence range; while (b) showcases a performance improvement spanning both intermediate and end convergence regions. Figure 14(c), on the other hand, is an example of the tg strategy (greedy strategy with thickened RS) consistently having the best performance. Among all these trials, MC samplings help to improve the performance of greedy strategies (light and dark blue) within the end convergence regions.

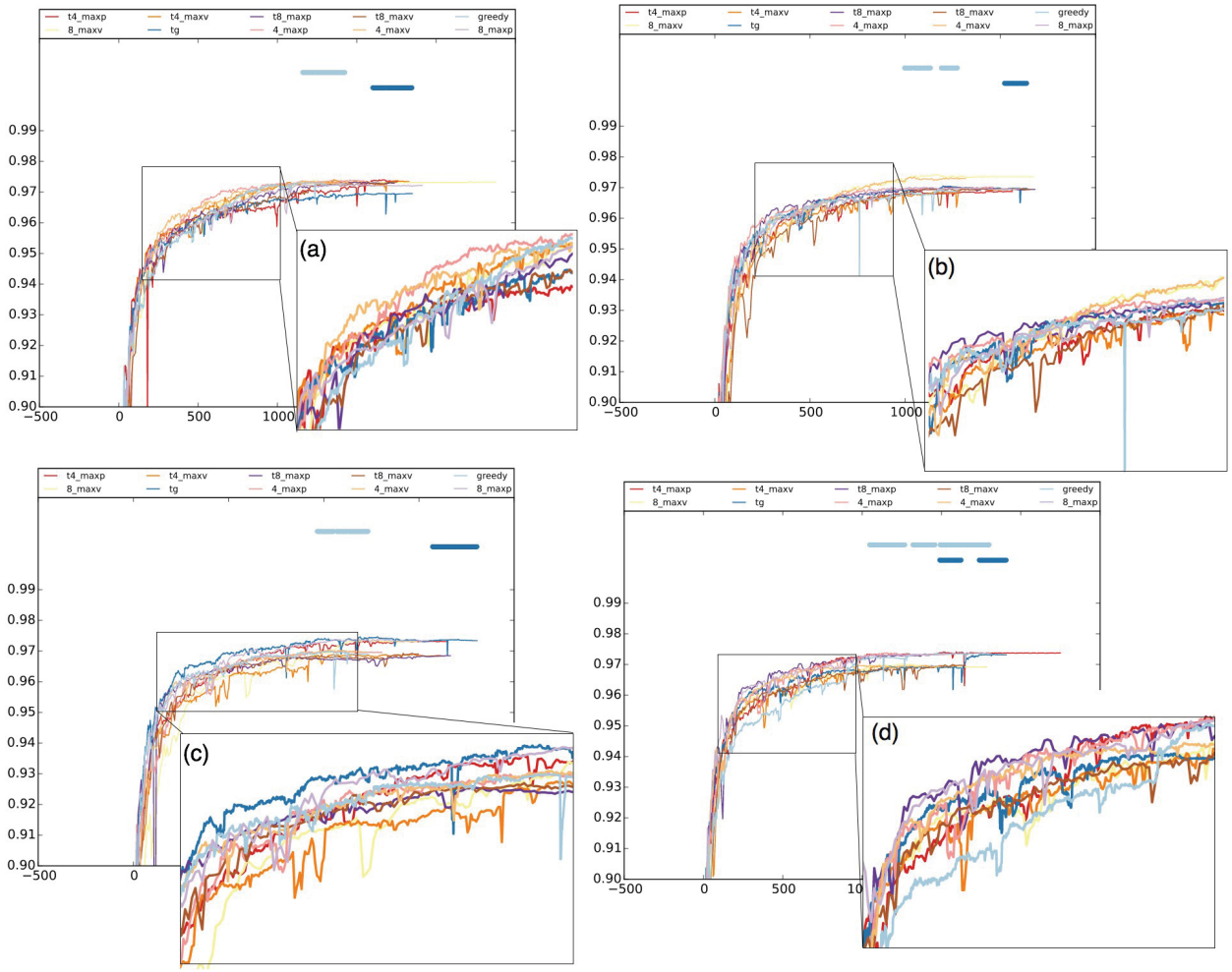


Figure 14. Convergence plots of the Hubble dataset using the weighted distance scoring with 10 processors.

Similar performance improvements using topology-inspired techniques are also observed for the Islands dataset under the same setting (Figure 15), where the topology-inspired batch selection techniques outperform the greedy strategies across the intermediate region in (a), both early and intermediate regions in (b), early region in (c) and almost all three convergence regions in (d). Finally, we obtain similar observations across the Face and Hubble datasets under the single processor setting and using the distance scoring (results omitted here).

In summary, the topology-inspired batch selection strategies can outperform the greedy strategy, in particular, within the intermediate convergence regions. This is not too surprising during this stage of the AS process, as an ROM is reasonably trained but would still benefit quite a bit from more exploratory strategies to improve its fit. Within the end convergence region, when an ROM is sufficiently well trained and only requires tiny amount of tweaking, exploratory strategies may lose their attractiveness. Furthermore, the topology-inspired strategies that outperform the greedy strategy vary across different trials. It would be a difficult task to try to predict what particular strategies would work well for a particular dataset. It depends on a fine balance between the exploitation and exploration, and relies on factors such as the initial training set, batch sizes, implementation details, and the topological complexities of the scoring functions and the reliability surfaces.

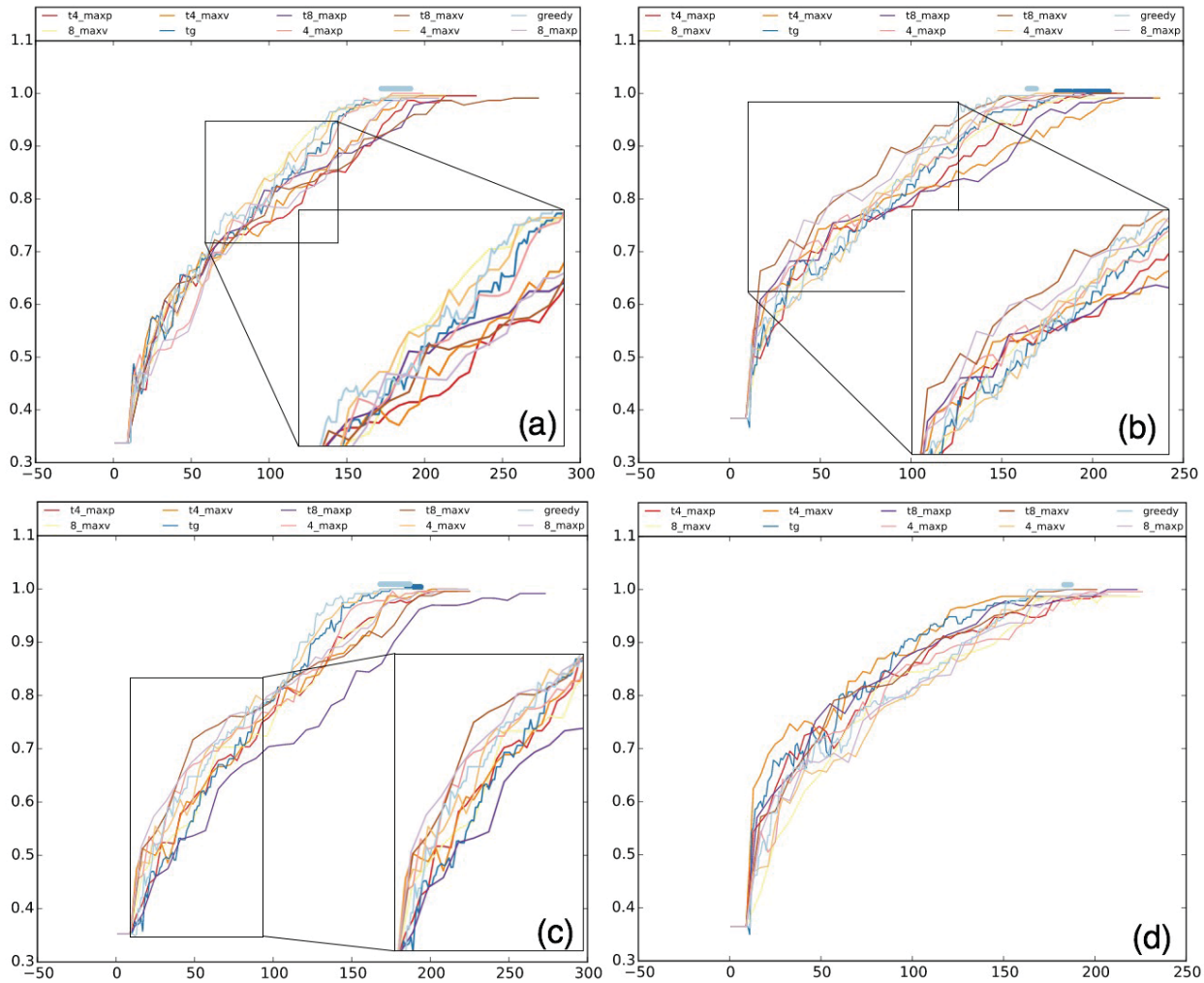


Figure 15. Convergence plots of the Circle and GMM datasets using the weighted distance scoring with 10 processors.

2.6.2.2 For simple RS, Topology-inspired Batch Selection does not Improve in Performance Against the Greedy Strategy. For an RS that contains very few connected components and has simple boundaries, for example, in the case of the Circle and GMM datasets (Figure 15), the greedy strategy typically performs comparably and some-times better than the topology-inspired batch selection strategies; then choosing the appropriate strategies relies on whether there exists a parallel labeling environment or models with slow training procedure (see Section 2). If these conditions are not suitable for the problem at hand, the greedy strategy might be a simpler and easier choice.

2.6.2.3 MC Sampling Occasionally Helps to "Unstick" the Greedy Strategy. As described in Observation A, the greedy strategy has limited exploratory ability and occasionally would get "stuck" in some locally optimal but globally unsatisfactory ROMs. That is, when the current training set does not contain any new information to improve the ROM, then the MC sampling is enabled within the RAVEN implementation. As illustrated in Figure 12 and Figure 13, MC sampling helps to explore drastically different regions of the domain when the greedy strategy gets stuck, therefore improves the overall performance.

2.6.2.4 RS Thickening Can Occasionally Help With the Domain Exploration. This is due to the added exploratory power using a slightly enlarged RS candidate set (see Figure 14(c) for an example). However, learning the right thickening parameter might be nontrivial.

2.7 Discussion

In this section, we review our exploratory effort in introducing batch selection, in particular, topology-inspired batch selection framework to RAVEN. We evaluate our proposed techniques over testing datasets and discuss their strengths and weaknesses. We demonstrate that for complex limit surfaces, topology-inspired techniques could help to explore the domain space more efficiently, in particular, within intermediate convergence regions. It would be interesting to further study their effects on performance during the early convergence regions where the exploration also plays an important role. Furthermore, since the MC sampling helps to improve the domain exploration when the AS process gets stuck at local optimal ROMs, it is possible that the best strategy would be a hybrid strategy involving the adaptive greedy strategy, adaptive (topology-inspired) batch selection and the MC sampling. Finally, the topology-inspired batch selection strategies described in this report are among a few examples of incorporating topology into active learning (e.g., Reference 17), and more generally, into the sensitivity analysis of nuclear datasets (e.g., Reference 18). We envision new techniques to be developed along these directions in the future by combining topological data analysis with machine learning techniques to enhance analysis and visualization capabilities within RAVEN.

3. THEORETICAL DEVELOPMENT AND IMPLEMENTATION RESULTS OF SURROGATE CONSTRUCTION ALGORITHMS FOR RAVEN'S MULTI-PHYSICS MODELS

3.1 Introduction

This section is dedicated to the theoretical development and implementation of surrogate construction algorithms for general multi-physics models. This work intends to support the application of reliability analysis methods to the coupled physics modules under RAVEN. Specifically, the first part of the work presented is concerned with development of reliable surrogate models for multi-physics, nonlinear, high-dimensional models. The second part focuses on implementation and demonstration of such algorithms to the MAMMOTH environment, coupling the neutronics model of RattleSnake and the fuel performance analysis of the BISON module. Theoretical background, associated algorithms, and their implementation details are discussed in this section of the report.

The utility of these surrogate models is essential to reliability analysis studies, because it allows exploration of the combined operational phase space and identification of the contours of the failure surface (i.e., limit surface and reliability surface). For this process to be successful, the theory and associated surrogate construction algorithms must address the primary challenges facing existing surrogate construction techniques, including the curse of dimensionality, the robustness of the predictions, and the ability to handle multi-physics models. Our goal is to develop a new algorithm in surrogate construction that is capable of upper-bounding its own prediction errors and employing efficient reduced order modeling techniques to address the high dimensionality of the models' interfaces and the multi-physics nature of their coupling. These capabilities are currently lacking in most of the state-of-the-art surrogate construction methodologies. The theory and algorithms developed here are expected to be invaluable to a wide range of U.S. Department of Energy initiatives, specifically those dealing with advanced modeling and simulation, safety analysis, uncertainty quantification, and sensitivity analysis; they all require access to highly accurate and fast-executing surrogate models.

The theoretical background and developments are reported here for a number of algorithms serving collectively as a framework for performing dimensionality reduction and surrogate model construction for general multi-physics models. In particular, we develop a basic dimensionality reduction algorithm for a

single physics model and describe the transformation between the original and reduced variables. Further, we explain how the errors resulting from reduction are quantified. Next, the dimensionality reduction algorithm is extended to a multi-physics setting by applying it at the interface between each two models. With this application, one can identify the effective dimensionality for the overall multi-physics model. Error analysis is also extended to the multi-physics case. Numerical experiments supporting the developed theory are also contained herein, which will demonstrate application to the MAMMOTH code system,[19] coupling RattleSnake and BISON codes for coupled neutron transport and thermal analysis calculations.

3.2 Background

Our overarching objective is to develop a fast-executing surrogate model with quantifiable error bounds. Any surrogate construction algorithm requires a training process where an assumed form for the surrogate model (i.e., a parametric expression such as a polynomial fit with unknown coefficients), is trained against samples generated by the original model.[20] This training process depends on the number of model parameters. Therefore, a key ingredient of our proposed approach is to perform an initial reduction in the number of model parameters prior to construction of the surrogate. The proposed reduction is different from a conventional screening approach that is used to identify a subset of the model parameters that are considered to have the most dominant impact on the model's responses of interest. The conventional screening approach is not expected to be feasible for a general multi-physics model with huge input and output data interfaces for each of the sub-physics models, which typically corresponds to state functions that are distributed in the phase space (i.e., the power distribution and fission rate density representing the input parameters to thermal analysis calculations). Performing a parametric study for this distribution is computationally infeasible.

Instead, we will adopt a dimensionality reduction approach, where a linear transformation of the parameters is identified via a range-finding algorithm (RFA).[21,22] The RFA identifies a small subset of effective parameters in the transformed space that can be used to describe the majority of response variations. Clearly, the conventional parametric screening approach may be thought of as one special case of the RFA approach, wherein the standard coordinate system is used to select the effective parameters. However, the RFA approach offers two advantages over the standard parametric screening approach, including the ability to identify an optimum transformation that minimizes the number of effective parameters while upper-bounding the errors resulting from discarding the rest of the parameter variations. The second advantage is key to ensuring the reliability of the reduction process, which is currently absent from conventional parametric screening approaches.

Accordingly, we adopt a two-step process for construction of the surrogate. The first step involves application of RFA to reduce the effective dimensions of the input and output interfaces for the respective sub-physics models. Once identified, construction of the surrogate model is recast in terms of the reduced dimensions.

In this background section, we first provide an overview of the RFA, which is considered to be the cornerstone of all subsequent algorithms developed here. In the following sections, we apply RFA to single physics models then extend it to multi-physics models, along with development of error metrics that bound the reduction errors. Finally, a surrogate construction algorithm is presented that is based on the reduced dimensions for the overall multi-physics model.

An RFA may be described as follows: consider a model described by a general nonlinear function f that relates n_x input parameters denoted by a vector $x \in \mathbb{R}^{n_x}$ to n_y output responses denoted by a vector $y \in \mathbb{R}^{n_y}$:

$$y = f\left(x\right) \tag{1}$$

and without loss of generality, we assume that this function passes through the origin (i.e., $y = 0$ when $x = 0$); this is possible via a simple translation of coordinates. The objective of the RFA is to identify two subspaces: one for input and another for output spaces, denoted respectively by two matrix operators, $\mathbf{K}_x$ and $\mathbf{K}_y$. The matrix $\mathbf{K}_z$ (where z denotes, respectively, x and y) is extremely rank-deficient (i.e., its rank is much smaller than its dimension):

$$\mathbf{K}_z \in \mathbb{R}^{n_z \times n_z} \text{ and } rank\left(\mathbf{K}_z\right) = r_z = n_z \tag{2}$$

These matrices are used to restrict the variations of the respective model interface variables (i.e., the input parameters and output responses) on hyperplanes (i.e., mathematical subspaces). This restriction may be described by a projection operation as follows:

$$x^{(r)} = \mathbf{K}_x x \text{ and } y^{(r)} = \mathbf{K}_y y \tag{3}$$

where the (r) superscripts denotes that the respective variables are constrained to a subspace. A rank-deficient matrix may be written using an orthogonal decomposition as follows:

$$\mathbf{K}_z = \mathbf{Q}_z \mathbf{Q}_z^T \tag{4}$$

where $\mathbf{Q}_z \in \mathbb{R}^{n_z \times r_z}$ is a skinny orthonormal matrix (i.e., the number of its columns is much smaller than the rows, with the columns being ortho-normal), with r_z columns representing a basis for a subspace, denoted hereafter by the active subspace

$$\mathbf{Q}_z = \left[\left[\mathbf{Q}_z\right]_{*1} \quad \left[\mathbf{Q}_z\right]_{*2} \quad \cdots \quad \left[\mathbf{Q}_z\right]_{*r_z}\right]$$

where $[\mathbf{Q}_z]_{*i} \in \mathbb{R}^{n_z \times 1}$ is the ith column of the matrix $\mathbf{Q}_z$. An active subspace for the input parameter space, by definition, spans all input parameter variations that have dominant impact on the output responses. Or in other words, the output responses are insensitive to parameter variations that are orthogonal to the active parameters subspace.

The proposed surrogate model takes advantage of this behavior by limiting parameter variations to the active parameters subspace. Similarly, an active subspace in the output space implies that the majority of the output variations is contained in the active response subspace, with the variations in the orthogonal complement of that subspace being very small; therefore, they can be discarded in construction of the surrogate model.

Next, we would like to distinguish here between the pre and post-reduction variables for the various models interfaces. For example, consider a generic z interface, where the original space has dimension n_z, implying that z has n_z degrees of freedom. The reduced variable $z^{(r)}$ also lives in the original space and has n_z components; however, its variation is constrained to a subspace of dimension r_z, implying that there are $n_z - r_z$ perfect correlations between its n_z components. The r_z components of z along the active subspace are described by another vector that lives in an r_z dimensional space, referred to as the active degrees of freedom of the variable z. This smaller vector is important because it will be used to construct the surrogate model.

The relationships between these variables are described by the following equations:

$$z^{(r)} = \mathbf{K}_z z, \quad z_{DOF}^{(r)} = \mathbf{Q}_z^T z, \text{ and } z^{(r)} = \mathbf{Q}_z z_{DOF}^{(r)} \quad . \tag{5}$$

The first equation implies $z^{(r)}$ is the projection of z along the active subspace, which is spanned by the columns of the matrix $\mathbf{Q}_z$. The second equation calculates the r_z components of $z^{(r)}$ along the active

subspace and aggregates them in a vector $z_{DOF}^{(r)}$. The last equation describes how the reduced variables may be reconstructed from the active Degree of Freedom DoFs. Note that in this reconstruction $z_{DOF}^{(r)}$ has r_z components, while $z^{(r)}$ has n_z components, which follows from the fact that $\mathbf{Q}_z$ has n_z rows and r_z columns.

Now, let us assume there is interest in building a surrogate model for the active DoFs of the output responses as a function of the active DoFs of the parameter space. This may be described as follows:

$$y_{DOF}^{SUR} = \tilde{f}\left(x_{DOF}^{(r)}\right) \tag{6}$$

where the tilde implies a parametric representation selected to approximate the original function *f*. The superscript "SUR" denotes the surrogate predictions, which requires some training to identify the "unknown" features of the surrogate model, often in the form of unknown coefficients. For example, consider that a second order polynomial is selected for the surrogate model, this may be described as follows:

$$y_{DOF}^{SUR} = \tilde{f}\left(x_{DOF}^{(r)}\right) = g^T x_{DOF}^{(r)} + x_{DOF}^{(r)T}\mathbf{H}x_{DOF}^{(r)} \quad . \tag{7}$$

This surrogate could be re-cast in terms of the original variables as follows:

$$y^{SUR} = \mathbf{Q}_y\left(g^T\mathbf{Q}_x^T x + x^T\mathbf{Q}_x^T\mathbf{H}\mathbf{Q}_x x\right) \tag{8}$$

where y^{SUR} is the surrogate predictions for the original variables. The goal of surrogate training is to determine the gradient vector *g* and the Hessian matrix **H**, whose elements represent the unknown coefficients of the surrogate model. Notice that *g* has only r_x components and **H** is r_x x r_x. If no reduction is done a priori, the number of unknown coefficients would be dependent on n_x (i.e., the original dimensionality of the parameter space). Also, note that in this brute force approach, each model response would be constructed using a separate surrogate model, implying that the total number of unknown coefficients would be proportional to n_y, which leads to intractable computational burden in terms of storage and data processing.

It is noteworthy to mention here that the proposed surrogate construction approached preceded by a dimensionality reduction introduces two sources of errors: the first source is due to the parametric form employed for the surrogate model and the second is due to the restriction of the parameter and response variables to their respective active subspaces. There is no general approach for quantification of the former source of error, especially when the model is treated as a black-box. An estimate for this error is typically done using the residual error of the regression process that is used to identify the unknown coefficients of the surrogate model. The latter source of errors can be quantified using the RFA.

Quantification of reduction errors is an essential requirement of the surrogate model developed here. To describe these errors, re-write the reduced model as follows:

$$\mathbf{K}_y y = f\left(\mathbf{K}_x x\right) \quad . \tag{9}$$

This representation implies that the original function remains unchanged and the reduction errors are resulting solely from constraining input parameter variations to the active parameter subspace and the output variations to the active response subspace. To describe these errors, we distinguish between errors resulting solely from reduction in the parameter space, the response space, and a combination thereof. First, consider the errors resulting from the input parameter space reduction:

$$e_x = \left\| f\left(x\right) - f\left(\mathbf{K}_x x\right)\right\| \le \varepsilon_x \quad \text{over all } x \in S_x \tag{10}$$

The e_x describes the errors in the output responses due to a reduction in the input parameter space, with ε_x representing an upper-bound for all possible x values. The second source of errors is due to reduction in the response space:

$$e_y = \left\| f\left(x\right) - \mathbf{K}_y f\left(x\right) \right\| \le \varepsilon_y \quad \text{over all } x \in S_x \tag{11}$$

This reduction implies that the responses are constrained to an active subspace assumed to contain the majority of response variations. Finally, the combined error for two simultaneous reductions in both the input parameter and output response spaces is given by:

$$e_{xy} = \left\| f\left(x\right) - \mathbf{K}_y f\left(\mathbf{K}_x x\right) \right\| \le \varepsilon_{xy} \quad \text{over all } x \in S_x \tag{12}$$

Earlier work has developed the theoretical background required for the quantification of these three error bounds. Interested readers may refer to Reference 22 for quantification of errors resulting from reduction in the input space, Reference 23 for reduction errors in the output space, and Reference 24 for the simultaneous reduction in both input and output space.

It is important to mention here that for a given selection of the active subspaces, one can create upper-bounds on the errors resulting from the reduction.[21] And for a given upper-bound on the error, one could find an infinite number of active subspaces that satisfy this upper-bound. The goal of the RFA is to identify an active subspace with the smallest possible rank in order to realize the benefits of the reduction which may be measured in terms of the cost required to build the surrogate model and the cost required for its execution. By way of example, reduction in the parameter space may be rendered via three approaches, gradient-free and gradient-based reduction, and a third hybrid approach.[25]

The gradient-free approach employs an upstream physics model to determine the active subspace for the parameters space of the downstream physics model. This approach is very similar to a proper orthogonal decomposition that has been well studied in many communities.[26] This is possible if the downstream model is used to calculate the parameters for the function f.

The gradient-based approach employs derivatives of the function f with respect to its parameters x to determine its active subspace.[22] For example, if f is a linear function, the gradient of f is selected as the optimum active subspace because all parameter variations that are orthogonal to the gradient will produce zero changes in the output responses. Earlier work has shown that for a general nonlinear function, the active subspace generated by applying the RFA algorithm to the gradient of f can be used to determine the parameters active subspace.[22]

Note that the gradient-free approach implies that the reduction rendered for a given physics model is determined by an upstream physics model; however, the gradient-based approach employs the given physics to determine its own reduction. Hybridizing both of these approaches would ultimately lead to a better reduction. Examples of such hybridization potential have been recently proposed, see for example References 27 and 28. We will, however, limit our algorithms development to the gradient-free approach only, as the physics models employed for demonstration (RATTLESNAKE and BISON) currently do not have capabilities to calculate the derivatives. Extension of the proposed algorithms using gradient-based reduction and its hybridization with gradient-free reduction will be left to future work.

Now, we turn to the mechanics of the RFA algorithms and how they are used to construct active subspaces for the parameter and response spaces. First, we discuss the construction of the response active subspace described as follows:

1. Specify the range of allowable x variations, contained in the volume $S_x \in \circ_x$.

2. Specify preset tolerance ε_y (i.e., upper-bound), on the reduction errors for the responses.

3. Generate $l+l_s$ random samples for x (i.e., $\{x_i\}_{i=1}^{l+l_s} \in S_x$).

4. Generate $l+l_s$ realizations of y, i.e., $\{y_i\}_{i=1}^{l}$ and form two matrices.

5. $$\mathbf{Y} = \begin{bmatrix} y_1 & y_2 & \dots & y_l \end{bmatrix} \in \mathbb{R}^{n_y \times l} \text{ and } \mathbf{Y}_s = \begin{bmatrix} y_{l+1} & y_{l+2} & \dots & y_{l+l_s} \end{bmatrix} \in \mathbb{R}^{n_y \times l_s}. \quad (13)$$

6. Compute $\mathbf{Y}_c$ and $\mathbf{Y}_s$ by standardizing the matrices $\mathbf{Y}$ and $\mathbf{Y}_s$, respectively.

7. Using $\mathbf{Y}_c$, compute orthonormal basis $\mathbf{Q}_y$ with rank r_y

8. Using $\mathbf{Y}_s$, assess whether $\mathbf{Q}_y$ satisfies the tolerance ε_y.

9. If the desired tolerance is not met, increase l.

The volume S_x in Step 1, defined by the user, identifies the allowable ranges for the parameters. This volume may be defined in a number of ways depending on the modeling conditions. For example, the simplest approach is to define an interval range for each of the parameters. If, for example, a parameter represents the concentration of a given material, then the interval can be selected to span the expected range of that concentration variation over the envisaged horizon of operation. If the parameter represents a technological quantity (i.e., a dimension subject to manufacturing uncertainty), then the range may be selected to cover the range of uncertainty expected (i.e., two or three standard deviations around its mean value). If the parameter represents a physical quantity (e.g., thermal conductivity, subject to a general aleatory or epistemic uncertainty), then the associated parameter probability density function is to be specified to sample the random parameter values in Step 3. If the parameters are measured experimentally in a manner that introduces correlations between their uncertainties (e.g., nuclear cross-sections), then the covariance matrix describing their uncertainties must be used to constraint their sampled values. Similarly, if the parameters are calculated from an upstream physics model, correlations between their variations are expected, and must be specified to ensure that the samples generated in Step 3 are consistent with the upstream physics model.

The y realizations in Step 4 may be generated directly using the model function, i.e., $y_i = f(x_i)$. If the function f is too expensive to evaluate, other approximate approaches may be used. For example, a lower fidelity model may be used (e.g., a diffusion model in lieu of a transport model, a deterministic in lieu of a probabilistic model). If y is obtained iteratively, the pre-converged values of y be used.[29] If y is too expensive to evaluate over the entire problem domain, the model may be restricted to a sub-domain (e.g., evaluate pin power distribution over a single fuel assembly vs. whole core).[30] While all these approaches introduce additional errors resulting in an increase in the size of the active subspace, it can be shown that the associated reduction errors can still be upper-bounded as described earlier.

Note that in Step 3 two sets of random samples are generated. The first set contains l samples which are used for the construction of the active subspace, referred to as the snapshots set. The second set contains l_s samples which are used solely for calculating an upper-bound on the active subspace in Step 7. This set is referred to as the oversamples set. It is important to distinguish between these two sets, because as mentioned earlier, the realizations used for the construction of the active subspace (i.e., the snapshots set), could be generated using an approximate model for the function f. However, the oversamples set must be calculated using the original function f to ensure reliable determination of the upper-bound. Secondly, the size of the snapshots set needs to be at least as big as the size of the sought active subspace; however the oversamples set may be set to a fixed value (typically less than 10), which are used to specify a probabilistic confidence in the estimation of the error upper-bound. Also note that if the snapshots set is too small to meet the user-defined tolerance on the reduction errors (Step 8),

additional snapshots must be added, however the oversamples set need not be re-evaluated or expanded, as it can be used to test multiple active subspaces.

Step 5 is particularly required when the components of the response y are of different units and/or scales. For example if y contains outputs from a typical core simulator such as the critical eigenvalue, pin powers, and the fuel isotopic composition, it is important to standardize y by centering it around and/or dividing it by the mean values of the samples. This is also important because the user-defined tolerances for the different components of the response vectors are expected to be different. For example, one desires to calculate the multiplication factor to five significant digits, while the power is acceptable to only three significant digits. The standardization process may be described as follows:

$$\mathbf{Y}_c = \mathbf{W}\left(\mathbf{Y} - \mathbf{Y}_0\right) \in \mathbb{R}^{n_y \times l} \tag{14}$$

where $\mathbf{Y}_0$ is a matrix with l identical columns $\left[\mathbf{Y}_0\right]_{*i} = y_0$, representing reference values for the responses, and $\mathbf{W}$ is a diagonal matrix used to convert the variations in relative values with appropriate weights reflecting their desired tolerances, i.e., $\mathbf{W}_{ii} = \tau_i / \left[y_0\right]_i$, where $\left[y_0\right]_i$ is the ith component of the y_0 vector, and τ_i is an additional weight. For example, if the ith and j^{th} responses are desired to d_i and d_j significant digits, respectively, then the weights may be selected as:

$$\log_{10}\left(\tau_i / \tau_j\right) = d_j / d_i \tag{15}$$

The same standardization process may be applied to $\mathbf{Y}_s$ containing the oversamples set.

In Step 6, an orthonormal basis for the active subspace is calculated. This may be done using any rank-revealing decomposition such as the singular value decomposition (SVD) or the Gram-Schmidt QR factorization, or any of their numerous variations.[31] This process generates a matrix $\mathbf{Q}_y$ with r_y columns. The premise is that this matrix can be used to reconstruct the model response realizations for any input parameter $x \in S_x$ such that the discrepancies between the original model responses and the reconstructed responses are upper-bounded by the user-defined bound ε_y .

This may be written as follows:

$$\left\|\left(\mathbf{I} - \mathbf{Q}\mathbf{Q}_y^T\right) f\left(x\right)\right\| \le \varepsilon_y \tag{16}$$

where $\mathbf{Q}_y^T f(x) \in \mathbb{R}^{r_y}$ represents the r_y components of the response along the active subspace (referred to as the responses active DoFs), and $\mathbf{Q}_y\mathbf{Q}_y^T f(x) \in \mathbb{R}^{n_y}$ is the re-constructed response vector in the response space. Note that the vector $\mathbf{Q}_y\mathbf{Q}_y^T f(x)$ has n_y components just like the original response vector $f(x)$. However, the variations of these n_y components is restricted to an r_y active subspace. Figure 16 depicts this situation for a 3D response space, and a 2D active subspace (representing a hyperplane).

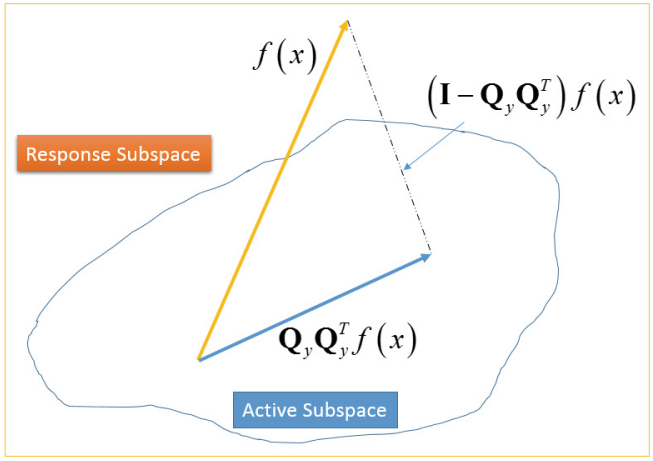


Figure 16: Active Subspace Reconstruction Error

The error upper-bound ε_y is calculated in Step 8 as follows:

$$\varepsilon_y = 10\sqrt{\frac{2}{\pi}}\max_{i=1,\ldots,s}\left\|\left(\mathbf{I}-\mathbf{Q}_y\mathbf{Q}_y^T\right)\left[\mathbf{Y}_s\right]_{*i}\right\| \text{ such that } p\left(e_y(x)\geq\varepsilon_y\right)=10^{-s} \tag{17}$$

where $\left[\mathbf{Y}_s\right]_{*i}$ is the ith column of the matrix $\mathbf{Y}_s$ and $e_y(x)$ is the error for any given x as defined in Equation (14).

This upper-bound is met in a probabilistic fashion, implying that there is a probability of 10^{-s} (s the size of the oversamples set) that the actual error $e_y(x)$ will exceed the bound ε_y for some parameter value x that belongs to the allowable range of parameter variations S_x. This probability is denoted as the failure probability (i.e., denoting the failure of the active subspace to upper-bound the errors resulting from the reduction). To implement this formula, the rank of the active subspace is increased progressively until the error tolerance is satisfied. The interested reader may find a proof for this upper-bound for linear operators in Reference 21, and its extension to smooth nonlinear operators in a multi-physics setting in Reference 25.

If the calculated upper-bound does not meet the preset user tolerance using the entire snapshots set, the size of the snapshots set must be increased by adding more model realizations. There is no general methodology for figuring out how many additional realizations should be added. However, some preliminary analysis of the error bound decline with the number of realizations could be used to estimate the rate of error reduction, which can provide a first order estimate of the required additional model realizations.[32]

To test the upper-bound generated by the RFA algorithm, a verification algorithm is applied:

1. Given specified preset tolerance ε_y assumed to be satisfied by an active subspace $\mathbf{Q}_y$, and a failure probability $p=10^{-s}$.
2. Generate l_v random samples for x, i.e., $\{x_i\}_{i=1}^{l_v}\in S_x$.
3. Calculate realizations using original model $\{y_i=f(x_i)\}_{i=1}^{l_v}$.

4. Form the standardized version of the verification matrix $\mathbf{Y}_v = \begin{bmatrix} y_1 & y_2 & \dots & y_{l_v} \end{bmatrix} \in \mathbb{R}^{n_y \times l_v}$.

5. Calculate the maximum errors for the verification set $\left\{ \delta_i = \left| \left(\mathbf{I} - \mathbf{Q}_y \mathbf{Q}_y^T \right) \left[\mathbf{Y}_v \right]_{*i} \right| \right\}_{i=1}^{l_v}$.

6. Calculate the fraction p_f of the verification set samples whose errors exceed ε_y.

7. If $p_f \leq p$, the verification test is positive.

In practice s (size of the oversamples set) is typically selected to render a very small failure probability p (i.e., 10^{-6} to 10^{-10}), which can only be observed numerically with large number of samples. Therefore, it is recommended to use a small number s to complete this test.[23]

Another utility of this test is that it allows one to generate an independent correlation between the size of the active subspace and the maximum error resulting from the reduction. This same correlation could be generated using the snapshots set, but for verification purposes, we employ the verification set since it is independent of the choice of the active subspace.

3.3 Range Finding Algorithms for Multi-Physics Models

We now turn to extending the RFA algorithm to a multi-physics model. The basic idea is to perform a dimensionality reduction at each model-to-model interface. Instead of presenting an abstract framework for a general multi-physics model, we elect to demonstrate the RFA algorithm to a specific multi-physics model that is currently being developed at INL, the RattleSnake-BISON coupled codes system under MAMMOTH.[19] This coupling employs RattleSnake to solve for the neutron field to calculate the power distribution, fission rate density, and accumulated fuel burnup that are fed as input to BISON to update the temperature field, which is then used in conjunction with burnup to update the macroscopic cross-sections for RattleSnake. The physics components comprising this coupling are depicted in Figure 17.

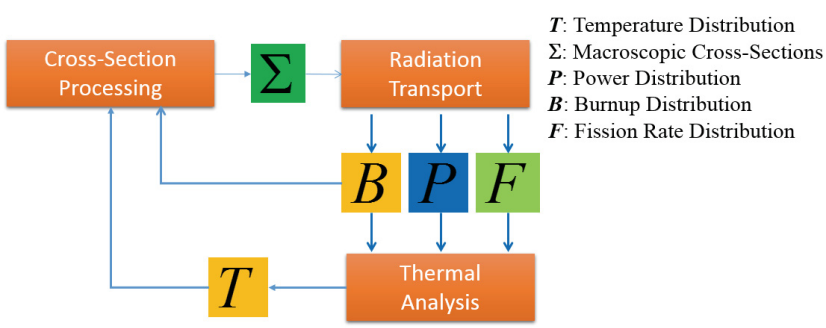


Figure 17. Multi-Physics MAMMOTH-based Model of RattleSnake and BISON.

In principle one can apply the RFA to each of these models to identify the degree of reduction that is inherent in each of the physics models. This knowledge could be used in a straightforward manner to construct a surrogate model for the respective physics models. This approach while simple to implement, it ignores the fact that effective DoFs associated with a given physics model may not be important to the other physics models. This implies that one would be constructing surrogate models that capture variations that may not be important to the overall model. Therefore, we adopt another approach in which the effective DoFs common to all physics models are first determined then employed to construct surrogates for the individual models. This approach has been shown in earlier work to be more effective as it identifies a smaller number of effective degrees of freedom, and hence is expected to be more computationally advantageous[28].

The above multi-physics model is mathematically described as follows:

$$\left[B,F,P\right]=f_{TR}\left(\Sigma\right),$$
$$T=f_{TH}\left(B,F,P\right),$$
$$\Sigma=f_{XS}\left(T,B\right) \tag{18}$$

where the subscripts *TR*, *TH*, and *XS* denote transport, thermal, and cross-section models, respectively. As described earlier, we will attempt to identify the minimum DoFs that influence the overall model. To achieve that, we need to develop notations that distinguish between different types of reductions introduced at each interface. In particular, we would like to distinguish between the reduction rendered by a single physics model, and the combined reduction of the physics model and its associated input parameters. In the first type of reduction, the RFA algorithm is applied by randomly sampling the parameter values over the entire parameter space, while in the second type of reduction, the parameters are sampled over an active subspace that is determined via reduction of the upstream physics model. Consider for example the thermal model, the following reduced outputs are identified as follows: $T^{(r)}$, $T^{(r)}_{(P)}$, $T^{(r)}_{(P,F)}$, and $T^{(r)}_{(P,F,B,\Sigma)}$. The $T^{(r)}$ implies that only the thermal physics model is used to reduce the dimensionality of the temperature distribution. This is achieved by sampling all the input parameters to the thermal model over their entire respective spaces. The $T^{(r)}_{(P)}$ implies that the sampled power values are confined to an active subspace, that is calculated by the transport model. In this case, the reduction rendered for the temperature distribution is due to both the thermal model and the input power distribution. Similarly, $T^{(r)}_{(P,F,B,\Sigma)}$ implies that the cross-sections are sampled over their corresponding active subspace, which subsequently generates a reduction for the fission rate, burnup, and power distributions, which are also used to confine their sampled values when applying the RFA to the thermal physics model. The idea of these multiple reductions is to identify the minimum number of degrees of freedom that are common to the overall multi-physics model. It is expected that the effective dimensionality of these various reductions to respect the following formula:

$$\mathrm{DOF}\left(T^{(r)}_{(P,F,B,\Sigma)}\right)\le\mathrm{DOF}\left(T^{(r)}_{(P)}\right)\le\mathrm{DOF}\left(T^{(r)}\right) \tag{19}$$

where DOF denotes the effective number of degrees of freedom that describe the variation of the reduced variables. Mathematically, this is equal to the effective rank of the matrix projection operator used to defined the reduced variables, i.e., $\mathrm{DOF}\left(T^{(r)}\right)=rank\left(\mathbf{K}_T\right)$, where $\mathbf{K}_T$ is defined in the sense of Equation (3) as $T^{(r)}=\mathbf{K}_T T$.

Ultimately, we should attempt to find the minimum number of degrees of freedom at each model-to-model interface to minimize the computational cost required to construct the surrogate model. Given the novelty of these proposed algorithms, we propose for implementation an explorative approach in which different levels of reductions are introduced to help the analysts develop their own insight into the mechanics of reduction algorithms. At the end, the best reduction for each model-to-model interface is identified, and denoted herein as: $X^{(r)}_{(0)}$, where *X* denotes the variable being reduced, and (0) denotes the best reduction.

Once identified, a surrogate model based on the reduced variables for each physics is constructed as follows:

$$\left[B^{SUR},F^{SUR},P^{SUR}\right]=\tilde{f}_{TR}\left(\Sigma^{(r)}_{(0)}\right),$$

$$T^{SUR} = \tilde{f}_{TH}\left(B_{(0)}^{(r)}, F_{(0)}^{(r)}, P_{(0)}^{(r)}\right),$$
$$\Sigma^{SUR} = \tilde{f}_{XS}\left(T_{(0)}^{(r)}, B_{(0)}^{(r)}\right) \tag{20}$$

The tilde denotes that the function is parametrically selected with unknown coefficients that are to be determined using training samples from the original models.

We now discuss the extension of the basic RFA algorithm to a multi-physics model, with the presentation customized here for the transport-thermal multi-physics model depicted in Figure 17.

1. Generate N_{XS} random realizations for the burnup and temperature distributions, $\left\{B_i, T_i\right\}_{i=1}^{N_{XS}}$, and calculate corresponding macroscopic cross-sections, $\Sigma_i = f_{XS}\left(B_i, T_i\right)$.
2. Calculate cross-section active subspace, $\mathbf{Q}_{XS} = \left[\ \left[\mathbf{Q}_{XS}\right]_{*1} \ \ \cdots \ \ \left[\mathbf{Q}_{XS}\right]_{*r_{XS}}\ \right] \in \mathbb{R}^{n_{XS} \times r_{XS}}$.
3. Calculate $\Sigma_i^{(r)} = \mathbf{K}_{XS}\Sigma_i$. The operator $\mathbf{K}_{XS}$ is defined below.
4. Execute the transport model, $\left[B_i, F_i, P_i\right] = f_{TR}\left(\Sigma_i^{(r)}\right)$
5. Calculate transport model output active subspace, $\mathbf{Q}_{TR} = \left[\ \left[\mathbf{Q}_{TR}\right]_{*1} \ \ \cdots \ \ \left[\mathbf{Q}_{TR}\right]_{*r_{TR}}\ \right] \in \mathbb{R}^{n_{TR} \times r_{TR}}$
6. Calculate $\left[B_i^{(r)}, F_i^{(r)}, P_i^{(r)}\right] = \mathbf{K}_{TR}\left[B_i, F_i, P_i\right]$.
7. Execute the thermal model, $T_i = f_{TH}\left(B_i^{(r)}, F_i^{(r)}, P_i^{(r)}\right)$
8. Calculate thermal model output active subspace, $\mathbf{Q}_{TH} = \left[\ \left[\mathbf{Q}_{TH}\right]_{*1} \ \ \cdots \ \ \left[\mathbf{Q}_{TH}\right]_{*r_{TH}}\ \right] \in \mathbb{R}^{n_{TH} \times r_{TH}}$
9. Calculate $T_i^{(r)} = \mathbf{K}_{TH}T_i$.

To gain insight into the reduction rendered by each physics model, the **K** operators (defined in steps 3, 6, and 9) at each model-to-model interface may be selected in three different manners. First, **K** is selected as the identity matrix implying that the randomized output realizations are passed directly to the next model without any reduction. This is equivalent to treating the combined physics models as one black box. This approach serves to identify the DoFs that affect the overall model. In practice however, this is not an efficient approach as the number of active DoFs may vary significantly from one physics model to the next. Therefore, it is important to perform some reduction at each model-to-model interface. This can be done by choosing **K** to be the projection operator calculated by the RFA, as described in Equation (4). This allows one to constraint the variations of the input parameters of a given physics model to the active subspace determined by the upstream physics model. To ensure that the active DoFs are equally excited by the RFA at each model-to-model interface, one can re-generate randomized samples for the downstream physics model, but now constrained to the active subspace determined by the upstream physics model. This can be done by selecting the perturbations as follows:

$$z_i^{(r)} = \mathbf{Q}_z z_{\mathrm{DOF},i}^{(r)} \ . \tag{21}$$

Applied to the cross-section interface, this becomes:

$$\Sigma^{(r)} = \mathbf{Q}_{XS}\Sigma^{(r)}_{\mathrm{DOF}}$$

$$\left[B^{(r)}, F^{(r)}, P^{(r)}\right] = \mathbf{Q}_{TR}\left[B^{(r)}_{DOF}, F^{(r)}_{DOF}, P^{(r)}_{DOF}\right] \quad . \tag{22}$$

Upon initial implementation of an RFA to a multi-physics model, it is recommended that these various selections for the **K** operators are performed to gain insight into the mechanics of the reduction as it applies to the particular physics application.

3.4 Surrogate Model Construction

Following the identification of the optimum active subspace at each model-to-model interface, any number of surrogate construction methodologies may be used, but now functionalized in terms of the active DoFs only. With the surrogates constructed in terms of the DoFs, transformation to the original spaces is straightforward using the **K** projection operators. Since the focus of this report is not on the exploration of different surrogate construction methodologies, we limit our discussion to a simple polynomial-type surrogate model. In future work, other forms of surrogate models will be investigated.

To determine the degree of the polynomial most adequate for each of the physics model, we employ an explorative approach in which the each of the physics models behavior (as measured in terms of its respective output active DoFs) is explored over randomized directions in its input active subspace. The algorithm below describes this process with demonstration to the transport model only. Application to other models is straightforward.

1. Let the input active DoFs of the cross-section space be given by: $\Sigma^{(r)}_{DOF}$
2. DO $I = 1,...,N_\beta$
3. Generate a random direction in the cross-section active subspace $\Sigma^{(r)} = \mathbf{Q}_{XS}\Sigma^{(r)}_{DOF}$
4. Generate N_α random points along the direction $\left\{\Sigma^{(r)}_i = \alpha_i \mathbf{Q}_{XS}\Sigma^{(r)}_{DOF}\right\}_{i=1}^{N_\alpha}$.
5. Execute the original transport model to calculate the responses $\left\{\left[B_i, F_i, P_i\right]\right\}_{i=1}^{N_\alpha}$
6. Calculate the active DoFs for the responses, $\left\{\left[B^{(r)}_{DOF,i}, F^{(r)}_{DOF,i}, P^{(r)}_{DOF,i}\right] = \mathbf{Q}_{TR}\left[B_i, F_i, P_i\right]\right\}_{i=1}^{N_\alpha}$
7. Plot each of the output active DoFs vs. the input active DoFs to determine trend
8. END DO *I*

This algorithm involves a loop, wherein a new random direction is generated in each iteration, and the function behavior is explored over that direction. One could plot the behavior of the responses of interest directly vs. the input active DoFs, which provides a visual aid describing the behavior of the physical quantities of interest. However, from a mathematical viewpoint, it is important to plot the behavior of the output DoFs since they are used as the basis for the surrogate model predictions. For example, if the output DoFs vary linearly with the input DoFs, then one has the assurance that any response of interest will also vary linearly with the input variations. The opposite however is not necessarily true. The loop repeats this process N_β times, i.e., generating N_β random directions in the input active subspace, to ensure a proper coverage of the space. If certain directions are associated with higher orders of nonlinearity, the proposed approach can detect these directions, which allows the user to increase the order of nonlinearity to capture model behavior for any possible parameter variation.

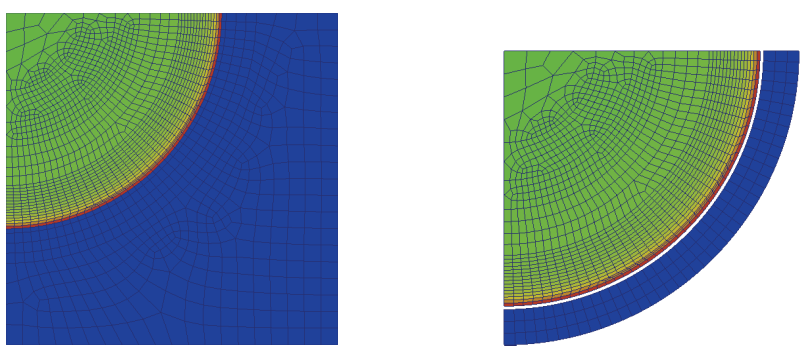
3.5 Numerical Results

The numerical experiment employs a 2D quarter pellet modeled by MAMMOTH[19] with the model parameters shown in Table 1. The numerical mesh used is described by 1667 finite elements and 4601 nodes values. The geometry and meshes are shown in Figure 17, which are generated by ParaView. Figure 17 (a) and (b) are the fine meshes employed, respectively, for the neutronics calculation in RattleSnake and the fuel performance calculation in BISON. Over the radial direction, the quarter pellet is divided into 23 radial rings, each with uniform material properties. The inner 20 rings represent the fuel materials, the 21st ring describes the gap, the 22nd ring for the cladding, and the 23rd region contains the water coolant/moderator. In this model, RattleSnake and BISON share the same mesh in fuel pellet part (i.e., fuel, gap, and cladding). The most outer part in RattleSnake mesh is water, which should be considered in neutronics calculation.

Table 1. Initial fuel pin parameters.

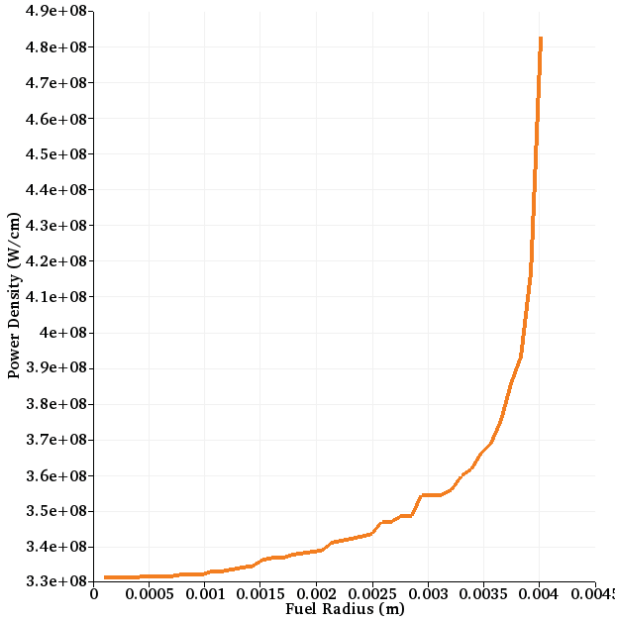
Parameter	Initial Value
Fuel Pin Radius	4.09575 (mm)
Gap Thickness	0.08255 (mm)
Outer Cladding Thickness	0.5715 (mm)
Pitch	12.5984 (mm)
Fuel Temperature	622 (K)

Power density, fission rate and burnup distributions, defined over the finite element mesh represent the output from RattleSnake, which are subsequently fed into BISON to calculate the temperature distribution over the nodes. The calculations are completed over 350 full power days with the reference results shown in Figure 19.

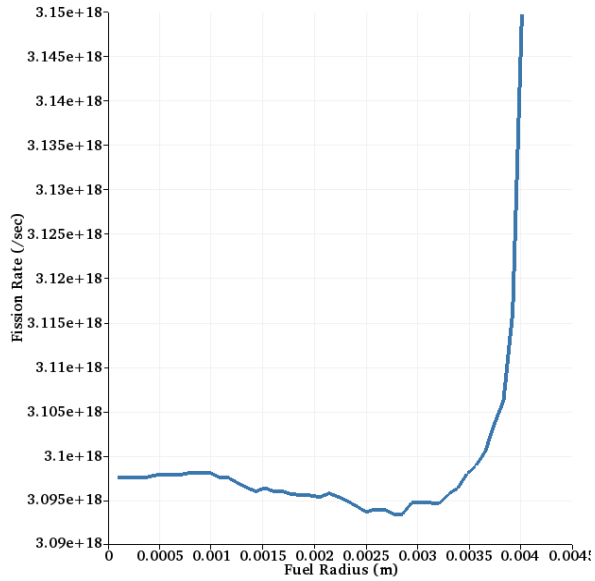


(a) RATTLES*N*AKE Mesh (b) BISON Mesh

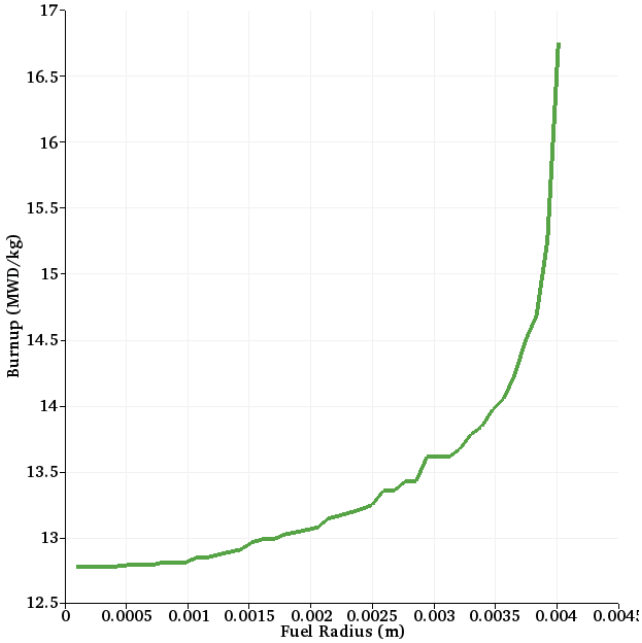
Figure 18. Numerical meshes employed by RattleSnake and BISON.



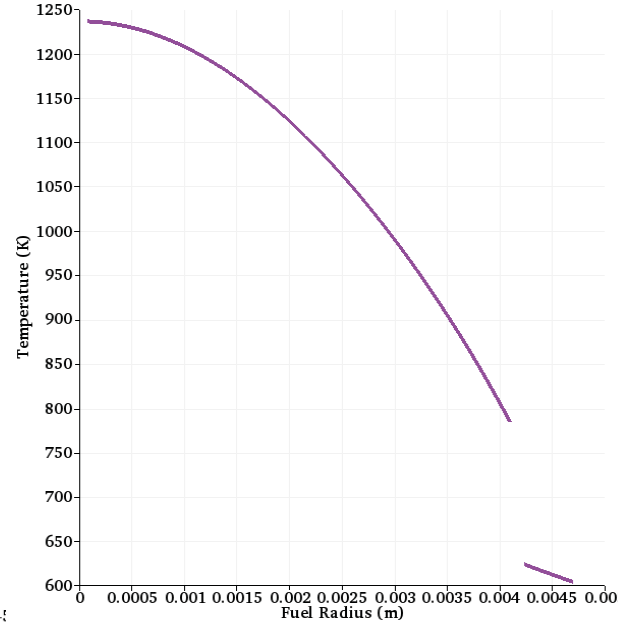
(a) Power Density Distribution



(b) Fission Rate Distribution



(c) Burnup Distribution



(d) Temperature Distribution

Figure 19. RattleSnake-BISON output distributions.

We now discuss the application of the RFA to the coupled calculations. Three different reduction scenarios are attempted to gain insight into the reducibility of the individual physics models of RattleSnake and BISON. Table 2 lists these different scenarios.

Table 2. Reduction scenarios.

Interface	Scenario 1	Scenario 2	Scenario 3
Cross-section	No, Σ	Yes, $\Sigma^{(r)}$	Yes, $\Sigma^{(r)}$
Power Density	Yes, $P^{(r)}$	Yes, $P^{(r)}_{(\Sigma)}$	Yes, $P^{(r)}_{(\Sigma)}$
Fission Rate	No, $F^{(r)}$	No, $F^{(r)}_{(\Sigma)}$	Yes, $F^{(r)}_{(\Sigma)}$
Burnup	No, $B^{(r)}$	No, $B^{(r)}_{(\Sigma)}$	Yes, $B^{(r)}_{(\Sigma)}$
Temperature	Yes, $T^{(r)}_{(P)}$	Yes, $T^{(r)}_{(\Sigma,P)}$	Yes, $T^{(r)}_{(\Sigma,B,F,P)}$

In the first scenario, no reduction is applied at the cross-section interface prior to the application of RFA to RattleSnake. The snapshots collected are used to reduce the power densities that are then subsequently used to generate random realizations for the BISON model. In doing so, the fission rate and burnup distribution are kept at their reference values. In the second scenario, the cross-section interface is reduced and the associated active subspace is used to constrain the random samples of the cross-sections. The third scenario repeats the second scenario but adds the reduction of the fission rate and burnup distribution.

The notations in the table are selected such as 'No' implies no reduction, while 'Yes' implies a reduction is rendered. The reductions applied are all reflected in the subscripts. For example, in scenario 2, the subscripts of the temperature distribution imply that both the cross-sections and power density interfaces were reduced with their associated active subspaces used to constrain the samples used in the RFA application.

In the first scenario, all cross-sections, corresponding to all material and reaction types, i.e., total, capture, fission, removal, absorption, nu-fission and kappa-fission cross-sections, for the 20 different material types available, are randomly perturbed within 5%. Snapshots of the power density, fission rate, and burnup distributions are collected, and their dimensionality is assessed using RFA.

All output distributions from RattleSnake are evaluated over the finite element mesh, containing 1667 elements. This means that the snapshot matrix $\mathbf{Y}_{TR}$ combining all distributions has dimensions of n_{TR}=1667x3 representing the number of rows, and l=7500 columns. The 7500 represent 500 snapshots (corresponding to 500 perturbations) collected over 15 depletion steps. By collecting the snapshots over the range of depletion one ensures that the reducibility is representative of the entire range of depletion expected during normal model execution. For the temperature distribution calculated by BISON, a matrix $\mathbf{Y}_{TH} \in \mathbf{R}^{4601\times 70000}$ of 200 snapshots (corresponding to 200 perturbations) is generated over the entire range of depletion, where the number of rows represent the number of nodes at which the temperature is calculated and the columns represent 200 perturbations, each containing 350 time steps.

The corresponding active subspace is calculated using SVD, and the associated column space of the active subspace is described by a matrix $\mathbf{Q}_{TR}$. This matrix corresponds to the left singular vectors of the SVD. To determine an appropriate rank for the active subspace, an estimate for the rank is gradually increased from 1 to 50, and in each time, the maximum relative error (error in the RattleSnake output distributions) is calculated according to Equation (11). The results are shown in Figure 20. These graphs allow one determine an appropriate rank that meets user-defined tolerances on all output distributions. Let this appropriate rank be given by r_{TR}.

The implication is that the first r_{TR} columns of $\mathbf{Q}_{TR}$, i.e.,

$$\left[\left[\mathbf{Q}_{TR}\right]_{*1} \quad .... \quad \left[\mathbf{Q}_{TR}\right]_{*r_{TR}} \right] \in \mathbb{R}^{n_{TR} \times r_{TR}} \tag{23}$$

can be used to bound the reconstruction errors for all output distributions to the tolerances defined by the user in the probabilistic sense of Equation (17). For example, considering only the power density with no cross-section reduction (first scenario), Figure 20 (b), and assuming a user-defined tolerance of 0.01%, one needs a rank of at least 30 to bound the reconstruction error to 0.01% with extremely high probability. A value of s=10 (in the sense of Equation (17) is used in this case, implying a failure probability of 10^{-10}.

The implication here is that despite that the power density in described over a spatial mesh of 1667 values and 15 in the temporal space, there are only 30 degrees of freedom that describes its variation in the combined space-time phase space.

In the second scenario, the cross-sections are reduced via an SVD decomposition applied on the raw cross-section data that are functionalized in terms of burnup, temperature, and material, and reaction types. The snapshot matrix is formulated such as each burnup value and temperature value and material type represents a new snapshot for the given cross-section. This follows because the cross-sections corresponding to different materials, and different compositions, and Doppler feedback are expected to be highly correlated. The results of this SVD are shown in Figure 20(a).

A rank of 20 corresponding to a maximum tolerance of approximately 0.01% for the cross-section reconstruction is assumed in the generation of the subsequent samples for RattleSnake. These samples are generated by constraining the cross-section randomized samples to the associated active subspace, described by $\left[\left[\mathbf{Q}_{XS}\right]_{*1} \quad .... \quad \left[\mathbf{Q}_{XS}\right]_{*20} \right] \in \mathbb{R}^{n_{XS} \times 20}$. The RFA is then used to execute RattleSnake 200 times to collect the power density, fission rate, and burnup snapshots.

The graphs on the right in Figure 21and Figure 22 are for the second scenario, while the left graphs describe the results of the first scenario. Note that in both scenarios, BISON is executed with randomized samples over the power density only, with the fission rate and burnup held constant at their reference values. Mathematically, this is described as follows:

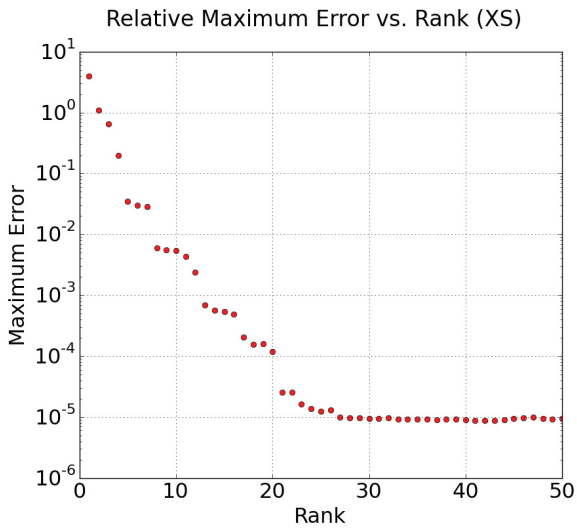
$$P_i^{(r)} = \mathbf{Q}_P \mathbf{Q}_P^T P_i \tag{24}$$

where P_i are the RFA randomized samples over the entire phase space of the power, and $\mathbf{Q}_P \mathbf{Q}_P^T$ is a projection operator on the active subspace of the power density.

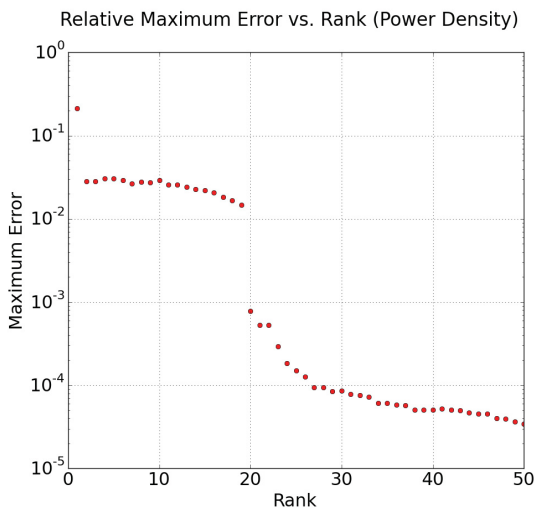
By limiting the RFA application to the power density only, we are able via the third scenario to explore whether the fission rate and burnup will introduce additional degrees of freedom in the temperature distribution. This follows because in the third scenario, all RattleSnake outputs are reduced and their associated active subspaces are used to constrain the RFA samples for BISON. Mathematically, this is described as follows:

$$\left[B_i^{(r)}, F_i^{(r)}, P_i^{(r)}\right] = \mathbf{K}_{TR}\left[B_i, F_i, P_i\right] \tag{25}$$

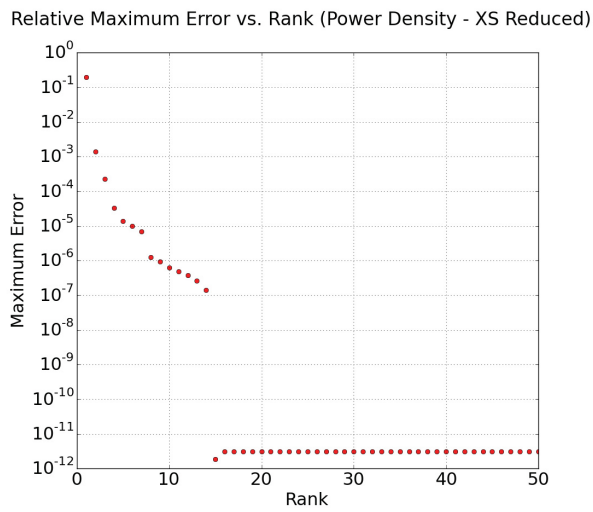
where $\mathbf{K}_{TR} = \mathbf{Q}_{TR}\mathbf{Q}_{TR}^T$.



(a) Error vs. Rank of Cross-Section Tables

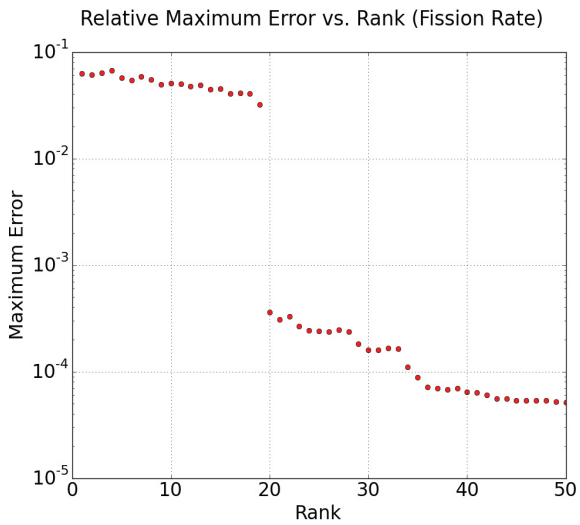


(b) Power Density

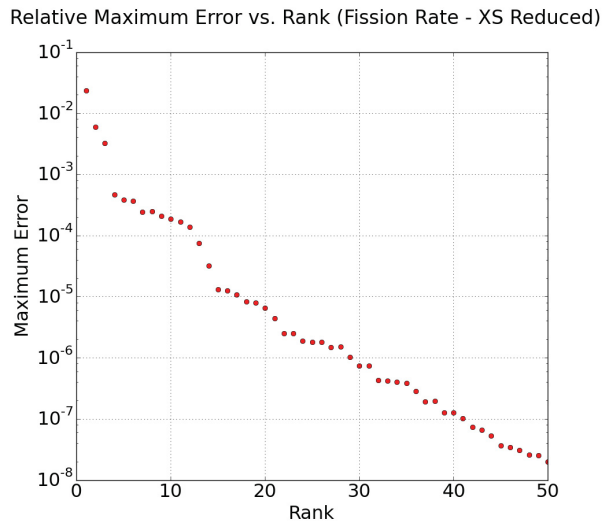


(c) Power Density (w/ XS Reduction)

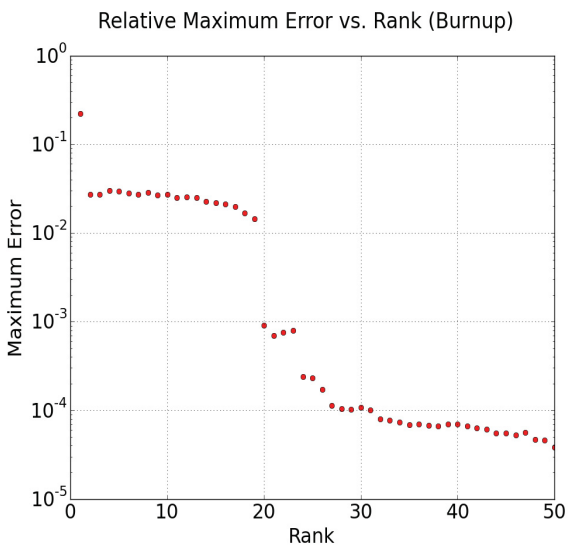
Figure 20. Maximum relative reduction error vs. size of active subspace for Scenarios 1 (b), 2 (c), and for the cross section active subspace (a).



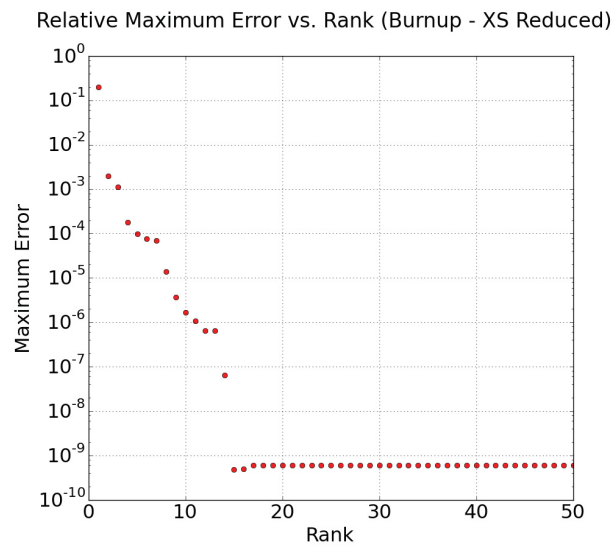
(d) Fission Rate



(e) Fission Rate (w/ XS Reduction)

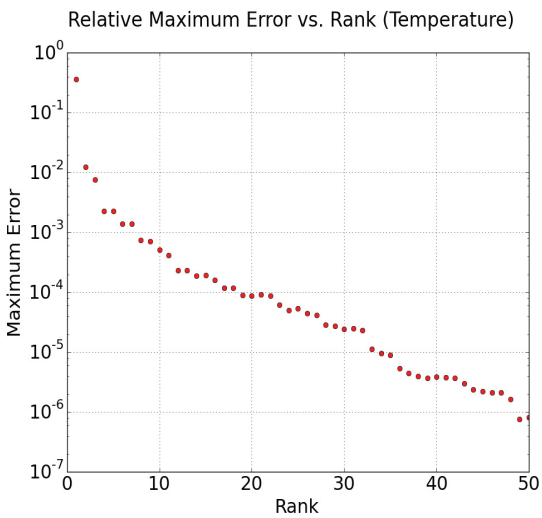


(f) Burnup

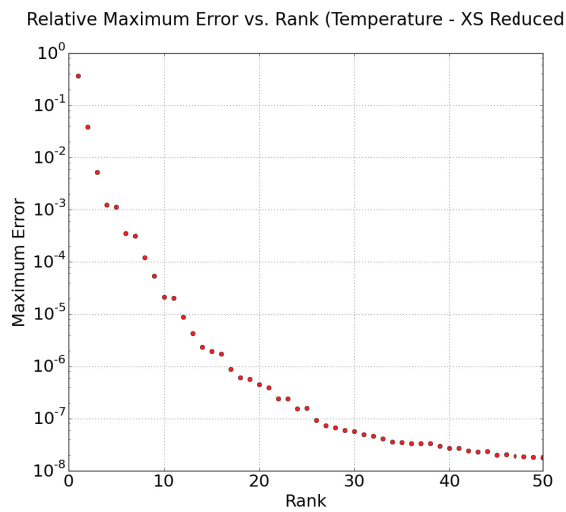


(g) Burnup (w/ XS Reduction)

Figure 21. Maximum relative reduction error vs. Size of active subspace for fission rate and burnup (on the left Case 1, on the right Case 2).



(h) Temperature



(i) Temperature (w/ XS Reduction)

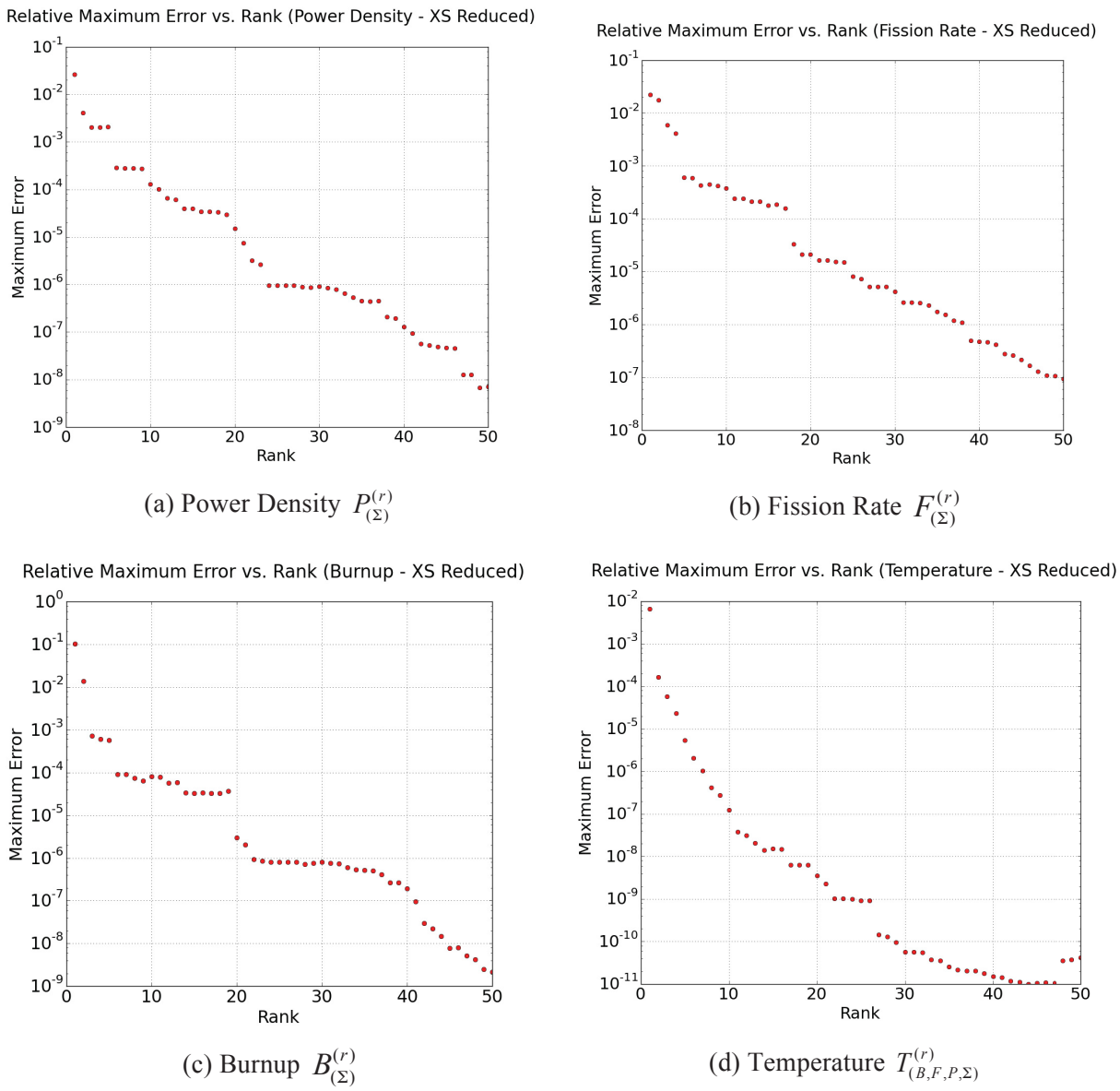
Figure 22. Maximum relative reduction error vs. size of active subspace for temperature (on the left Case 1, on the right Case 2).

A quick glance over the graphs in Figure 20-Figure 22 shows that one can consistently get a smaller active subspace in the second scenario when compared to the reduction results of the first scenario. For example, the power density needs an active subspace of size 30 to achieve an error tolerance of 10^{-3} in the first scenario, which drops by order of magnitude to only 3 with cross-section reduction. Notice that at rank 4, Figure 20(c), the error drops below 10^{-3}. Similar behavior may be observed for the fission rate and burnup distributions. The temperature distribution shows only a slight reduction in the size of the active subspace going from rank 8 to rank 6 for an error tolerance of 10^{-3}. This indicates that the majority of the reduction is taking place in the cross-section space, and the neutronics calculations. Also, notice that in the second scenarios, the temperature rank is about 6, while the power density is only 3. This implies the fission rate and burnup distributions have introduced additional degrees of freedom. Note that while the fission rate is expected to be highly correlated with the power density, since the only difference is the energy released per fission. This energy however is isotope dependent with a range of variation of approximately 5 to 10%. Therefore, if one requires a tight tolerance on the reduction results, one should expect additional DoFs to be introduced to capture this range of variations for the isotope-dependent fission energy release.

Finally, in the third scenario, all interface variables calculated by RattleSnake are included in the RFA application. This implies the active subspace is calculated for all three variables combined, which may be achieved by simply stacking the snapshots matrix for each variable in one bigger matrix.

$$\mathbf{Y}_{TR} = \begin{bmatrix} B_1 & \dots & B_{N_S} \\ F_1 & \dots & F_{N_S} \\ P_1 & \dots & P_{N_S} \end{bmatrix} \in \mathbb{R}^{5301\times 7500} \tag{26}$$

Notice that the number of rows here is simply 3 times the number of finite element meshes, since all three variables are defined on the same finite element mesh. The active subspace corresponding to this bigger matrix is calculated and used to re-evaluate the error as a function of the size of the active subspace. The results are shown in Figure 23.



(a) Power Density $P_{(\Sigma)}^{(r)}$

(b) Fission Rate $F_{(\Sigma)}^{(r)}$

(c) Burnup $B_{(\Sigma)}^{(r)}$

(d) Temperature $T_{(B,F,P,\Sigma)}^{(r)}$

Figure 23. Maximum relative reduction error vs. size of active subspace (Case 3).

Comparison of the graphs in Figure 23 to the right graphs in Figure 20 through Figure 22 show a slight increase in the size of the active subspace for the power density and fission rate is noticed. This is due to the reason given before regarding the slight variations in the isotope-dependent energy release per fission. It is also noticed that the burnup has only rank 2. Finally, the temperature distribution has a rank of 2 for an error tolerance of 10^{-3} and rank 1 for tolerance of 10^{-2}. This indicates that the temperature distribution variations over the range of depletion is highly correlated with the total energy production. This follows since the temperature variations are controlled by thermal parameters such as the thermal conductivity, heat capacity, etc., which are functions of the amount of radiation damage inflicted on the fuel, which is a function of accumulated energy deposited in the fuel over the irradiation horizon.

The final task of this part of the project is to construct a surrogate model for the coupled RattleSnake-BISON models. We achieve that by functionalizing the surrogate model in terms of the active DoFs of the various interfaces. We recognize that a plethora of algorithms exist in the literature for the construction of surrogate models. We will focus here on a simple polynomial fitting function as the main goal here is to

demonstrate that significant reduction in the cost of the surrogate model could be achieved via a priori dimensionality reductions of the model interfaces. Future work could be invested to determine the optimum surrogate model format.

To determine the degree of the polynomial needed, we perform a parametric study to explore the expected variations of the model responses with respect to the active DoFs. We assume a surrogate model of the form:

$$T^{SUR} = \tilde{f}\left(\Sigma\right) \tag{27}$$

which implies for a given cross-section variations, one can determine the corresponding temperature distribution over space and time. This surrogate is composed of four sub-surrogate models, described as follows:

$$\begin{aligned}
&T^{SUR} = \mathbf{K}_{TH} T_{DOF}^{(r)SUR}, \\
&T_{DOF}^{(r)SUR} = \tilde{f}_{TH}\left(B_{DOF}^{(r)SUR}, F_{DOF}^{(r)SUR}, P_{DOF}^{(r)SUR}\right), \text{ and} \\
&\left[B_{DOF}^{(r)SUR}, F_{DOF}^{(r)SUR}, P_{DOF}^{(r)SUR}\right] = \tilde{f}_{TR}\left(\Sigma_{DOF}^{(r)}\right) \\
&\Sigma_{DOF}^{(r)} = \mathbf{Q}_{XS}^{T}\Sigma
\end{aligned} \tag{28}$$

Taken from the bottom up, the active subspace of the cross-sections is used to project the user-defined cross-section variations (which could have as many as n_{XS} DoFs) onto a subspace with r_{XS} DoF(i.e., $\Sigma_{DOF}^{(r)} \in \mathbb{R}^{r_{XS}}$). The $\Sigma_{DOF}^{(r)}$ DoFs are subsequently used to calculate via the surrogate function $\tilde{f}_{TR}$ the corresponding variations in the DoFs of the fission rate, burnup, and power density variations. These variations are then used to calculate the variations in the DoFs of the temperature distribution $T_{DOF}^{(r)SUR}$ via the surrogate function $\tilde{f}_{TH}$. Finally, the projection operator $\mathbf{K}_{TH}$ is used to map the variations in the temperature DoFs to the full phase space of the temperature distribution.

An important part of the surrogate construction is to explore the appropriate shape for the functions $\tilde{f}_{TH}$ and $\tilde{f}_{TR}$ that is needed to capture the real variations of the BISON and RattleSnake models. Since these functions relates active DoFs, we execute the codes by perturbing their DoFs using a conventional parametric approach, wherein one parameter, i.e., one active DoF, is perturbed as a time, and the model behavior is visually inspected to determine the best polynomial function. Figure 24 shows one typical result showing the dependence of the most dominant temperature active DoF on the most three dominant DoFs of the power density. Similar results can be obtained for the other DoFs of the temperature distribution and the fission rate and burnup density. These results indicate that a linear trend is acceptable for this multi-physics model. The implication is that the surrogate functions could be described using matrix operators. Therefore, the final surrogate model may be described by a single matrix operator of the form:

$$T^{SUR} - T_{ref} = \mathbf{F}^{SUR}\left(\Sigma - \Sigma_{ref}\right) \tag{29}$$

where *ref* denotes some reference conditions, and $\mathbf{F}^{\mathrm{SUR}}$ lives in $\mathbb{R}^{n_{TH} \times n_{XS}}$ phase space but has only rank r (assumed to be the optimum rank for the overall multi-physics model), and can be written as:

$$\mathbf{F}^{SUR} = \mathbf{Q}_{TH}\mathbf{R}\mathbf{Q}_{XS}^{T}, \tag{30}$$

and $\mathbf{Q}_{TH} \in \mathbb{R}^{n_{TH} \times r}$, $\mathbf{Q}_{XS} \in \mathbb{R}^{n_{XS} \times r}$, and $\mathbf{R} \in \mathbb{R}^{r \times r}$. As discussed earlier, both the Q matrices are calculated from the initial dimensionality reduction process, implying that the surrogate construction involves only the construction of a small r x r square matrix, which may be obtained via least squares. For this study, a value of $r = 2$ is selected to keep the error tolerance for the temperature distribution below 10^{-3}. Given the linearity of the surrogate model, a rank of 2 implies that the temperature distribution have only two modes of variations, i.e., active DoFs. Similarly, the other model interfaces, i.e., power density, fission rate, burnup, and cross-section have two active DoFs that control the two modes of temperature variations, which are readily calculated from the RFA results. For example, Figure 25 shows the first two dominant modes for the power density.

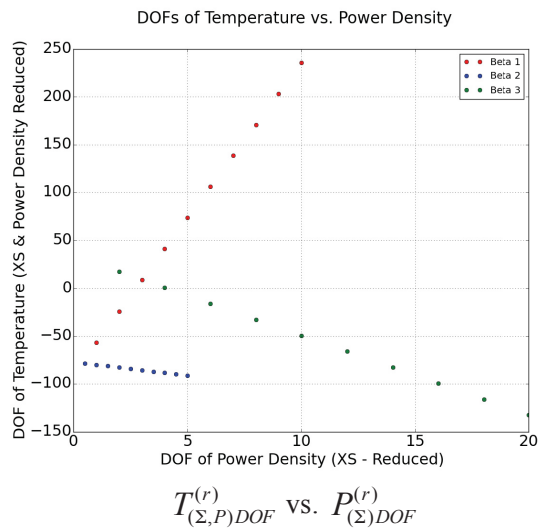


Figure 24. Relationship between the DoFs of RattleSnake and BISON.

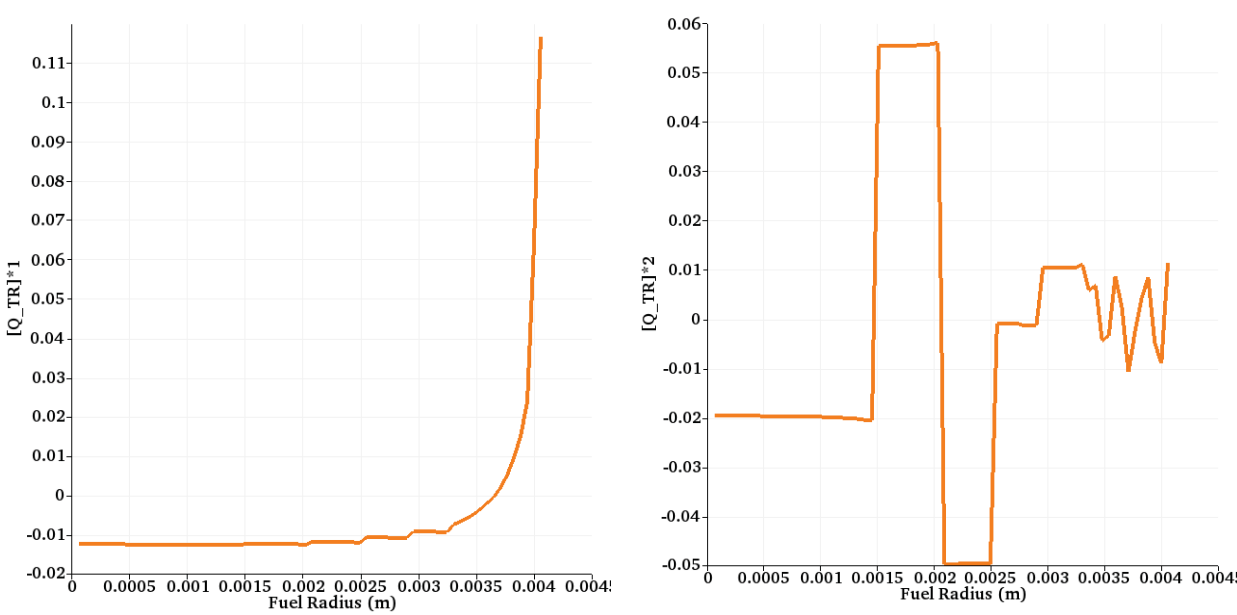


Figure 25. First two DoFs of power density.

4. META-MODEL IMPLEMENTATION IN RAVEN FRAMEWORK

4.1 Introduction

RAVEN is currently able to construct multi-targets reduced order models[10], which are aimed to represent the response of a system (in a fixed configuration) for multiple figures of merit and time-dependent ROMs[33]. These capabilities represent the initial steps for a larger implementation about the interaction of multiple models. In fact, in several cases, multiple models need to interface with each other since the initial conditions of one are dependent on the outcomes of another.

To better understand the problem that here is solved, it is useful to consider two simple examples:

1. The following problem is considered: a weather forecast simulation code "a" is used to compute the external (i.e., ambient) temperature in a certain location. A second model "b" is inquired to compute the average temperature in a room having as boundary condition, among several others, the external ambient temperature. The response of the model "b" depends on the outcome of the model "a."
2. Two different simulation codes are considered: (a) a code that is meant to compute the thermal conductivity of the ceramic uranium dioxide as function of the Temperature, and (b) a Thermal-hydraulic code that is used to compute the Temperature field of a reactor, whose heat conduction depends on the thermal conductivity value. As easily inferable, the two models are mutual dependent, leading to a non-linear system of equations.

The two reported examples are only aimed to illustrate the reason why the creation of a framework to make interact different models is a key development for the advancement of RAVEN as a comprehensive calculation flow driver. Before reporting how the meta-models have been implemented, it is necessary to briefly describe the representative Model "entities" that are available in RAVEN.

4.2 Models in RAVEN

The Model entity, in the RAVEN environment, represents a "connection pipeline" between the input and the output space. The RAVEN framework does not own any physical model (i.e., it does not posses the equations needed to simulate a generic system), but implements Application Programming Interfaces by which any generic model can be integrated and interrogated. In the RAVEN framework four different model categories (entities) are defined:

- Codes
- Externals
- ROMs
- Post-Processors.

The *Code* model represents the interface object that establishes the communication pipe between RAVEN and any driven code. Currently, RAVEN has Application Programming Interfaces for several different codes:

- RELAP5-3D, the most widely used Safety Code (thermal-hydraulic)
- RELAP-7, safety code eventual future replacement of RELAP5-3D code
- Any MOOSE-based application
- SAS4A/SASSYS-1, safety analysis code for fast reactors (Argonne)
- Modelica, object-oriented, declarative, multi-domain modeling language for component-oriented modeling of complex systems

- MELCOR, engineering-level computer code that models the progression of severe accidents in light-water reactor nuclear power plants (coupling under development by the University of Rome "La Sapienza")
- MCNP, general-purpose Monte Carlo N-Particle code that can be used for neutron, photon, electron, or coupled neutron/photon/electron transport (under development).

The data exchange between RAVEN and the driven code can be performed either by direct software interface or by files. If the system code is parallelized, the data exchanging by files is generally the way to follow since it can be much more optimized in large clusters.

The External model allows the user to create, in a Python file (imported, at run-time, in the RAVEN framework), its own model (e.g., set of equations representing a physical model, connection to another code, control logic, etc.). This model will be interpreted/used by the framework and, at run-time, will become part of RAVEN itself.

The ROM represents an Application Programming Interface to several different algorithms. A ROM is a mathematical representation of a system, used to predict a selected output space of a physical system. The creation and sub-sequential usage of a ROM involves a procedure named "training." The "training" is a process that uses sampling of the physical model to improve the prediction capability (capability to predict the status of the system given a realization of the input space) of the ROM. More specifically, in RAVEN the ROM is trained to emulate a high fidelity numerical representation (system codes) of the physical system.

The post-processor model is aimed to manipulate the data generated, for example, employing a sampling strategy. In RAVEN several different post-processors are available: (1) statistics post-processor, aimed to compute all the statistical figure of merits (e.g., expected values, variance, skewness, covariance matrix, sensitivity coefficients, etc.); (2) reliability surface, which computes the Limit Surface, inquiring a goal function (i.e., a function that determines if a certain coordinate in the input space led to a failure or success), and so many others.

4.3 Meta-Model in RAVEN

As already mentioned, in several cases multiple models need to interface with each other since the initial conditions of some are dependent on the outcomes of others. In order to face this problematic in the RAVEN framework, a new model category (e.g., class), named *MetaModel*, has been implemented. This class is able to assemble multiple models of other categories (i.e., code, external model, ROM), identifying the input/output connections, and, consequentially the order of execution and which sub-models can be executed in parallel.

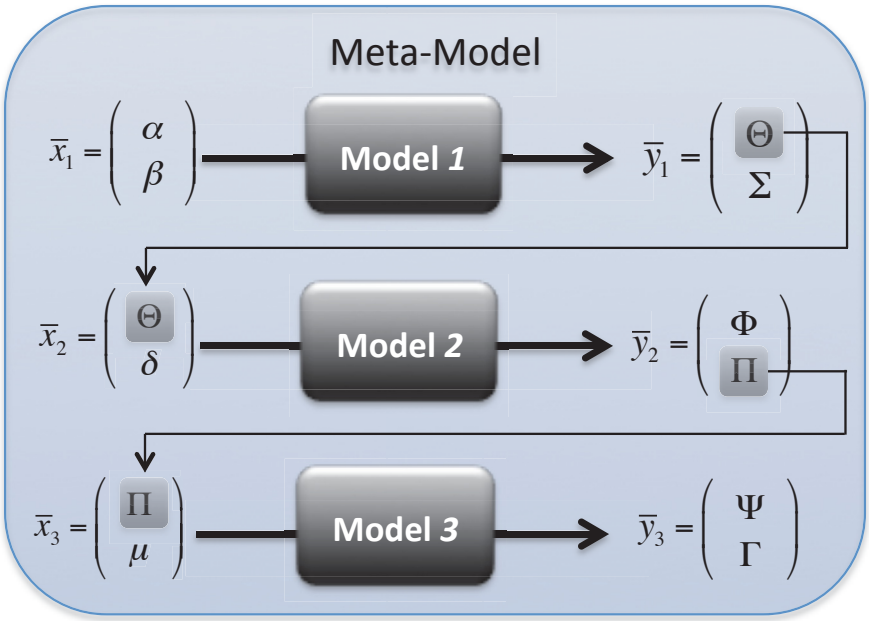


Figure 26. Example of a meta-model constituted by three sequential sub-models.

Before analyzing in detail how the meta-model capability has been developed in the RAVEN framework, it is worth to report a couple of schematic cases that show how the input/output interconnections determine the order of execution of the sub-models. Figure 26 reports an example of a meta-model that is constituted by three sub-models (ROMs, codes, or external models). As it can be noticed:

- *Model 2* is connected with *Model 1* through variable Θ (Model 1 output and Model 2 input)
- *Model 3* is connected with *Model 2* through variable Π (Model 2 output and Model 3 input).

In this case, the meta-model is going to drive the execution of all the sub-models in a serial sequence, because each model (except the *Model 1*) is dependent on one of the outcomes of the previously executed model.

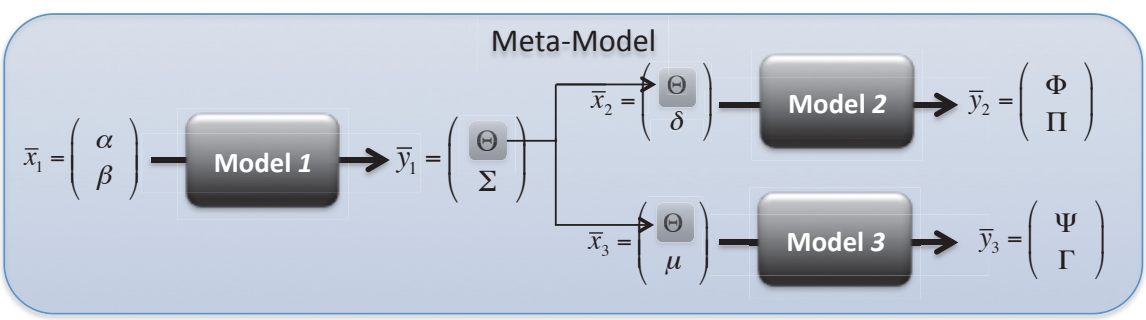


Figure 27. Example of a meta-model constituted by three sub-models, two of which can be run independently.

In Figure 27, another example is reported. In this case, *Model 2* and *Model 3* depend on *Model 1* only through the output variable Θ and they are not connected between each other. For this reason, the meta-model executes *Model 2* and *Model 3* in parallel, after inquiring about *Model 1*.

4.3.1 Implementation

Since modularity of the RAVEN framework, implementation of the meta-model entity has been straightforward. In RAVEN, all models' outputs (e.g., whatever code output) are collected in internal containers (named *DataObjects*) that are aimed to store time-series and input/output data relations in a standardized fashion; in this way, the communication of the output information among different entities (i.e., models) can be completely agnostic with respect to the particular type of output generated by a model. The meta-model entity fully leverages this peculiarity in order to transfer the data from a model to the other(s).

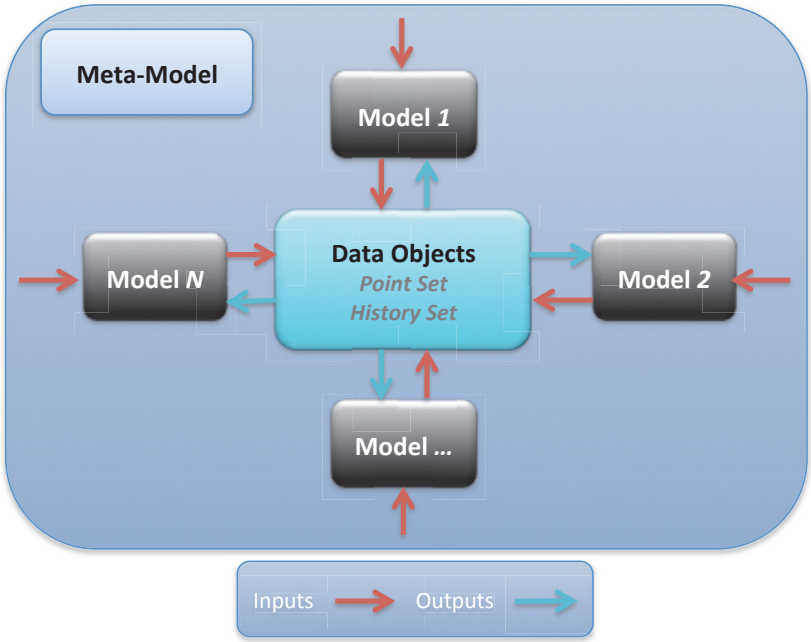


Figure 28. Meta-model data exchange among sub-models.

Based on the input/output relations of each sub-model, the meta-model entity constructs the order of their execution and, consequentially, the links among the different entities.

Figure 28 schematically shows the communication piping established by the meta-model entity. It can be noticed how the sub-models share information (inputs/outputs data) using the *DataObjects* entity as communication network.

4.3.2 Meta-model Resolving in a Non-Linear System

In several cases, the input of a model depends on the output of another model whose input is the output of the initial model. In this situation, the system of equation is non-linear and an iterative solution procedure needs to be employed. The meta-model entity in RAVEN is able to detect the non-linearity of the sub-models' assembling and activate the non-linear solver: Picard's iterative scheme. Because Picard's iterative scheme is well known, there is no mean to report its implementation here. Figure 29 shows an example of when the meta-model entity activates the Picard's iteration scheme, which ends when the residue norm (between an iteration and the other) falls below a certain input-defined tolerance.

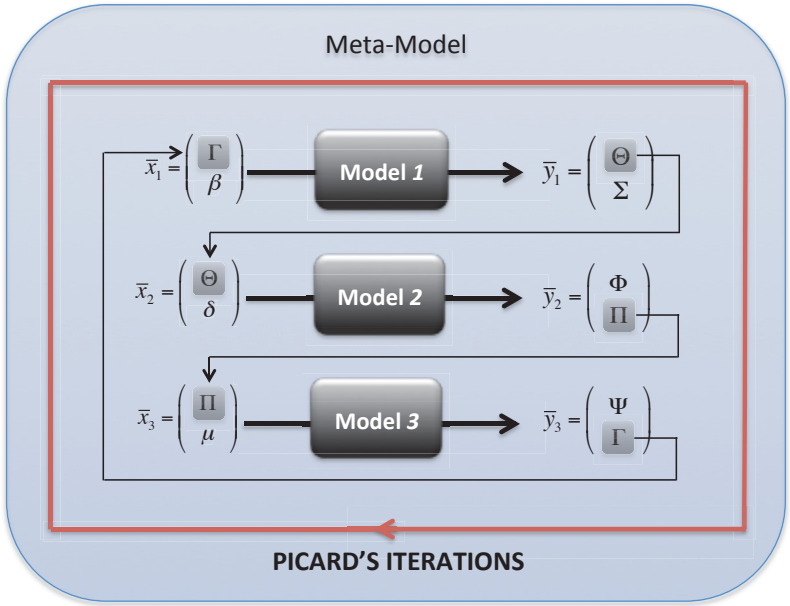


Figure 29. Meta-model resolving in a non-linear system of equations – Picard's iterations.

4.4 Application Example

In order to test the newly developed capability, a simple one-dimensional heat conduction transient (in a slab of thickness L=1 m) has been employed. Two external models compose the meta-model:

1. *EM1*: is aimed to solve the heat conduction partial differential equation:

$$\begin{cases} \frac{dT(x,t)}{dt} = k\frac{d^2T(x,t)}{dx^2} \\ T(0,t) = T_{left} \\ T(L,t) = T_{right} \end{cases} \tag{31}$$

2. *EM2*: computes the thermal conductivity (input of *EM1*) as function of the average temperature in the slab, using the following correlation:

$$k = \frac{38.23}{129.2+T} + 6.077E^{-13} * T \tag{32}$$

The transient is 10 seconds long and the initial temperature across the slab is set to 600 K. The so-constituted meta-model has been sampled through a grid strategy sampling the boundary conditions $T(0,t)$ and $T(L,t)$ with a uniform probability distribution between 500 and 1700 K.

Two cases have been run in order to test the functionality of the meta-model when it resolves in a chain of evaluations and in a non-linear system:

1. In the first test the model *EM2* uses the sampled boundary conditions to calculate the average temperature and consequentially the thermal conductivity that is fed in the heat conduction model *EM1*. This approach resolves in a chain of evolutions as shown in Figure 30.

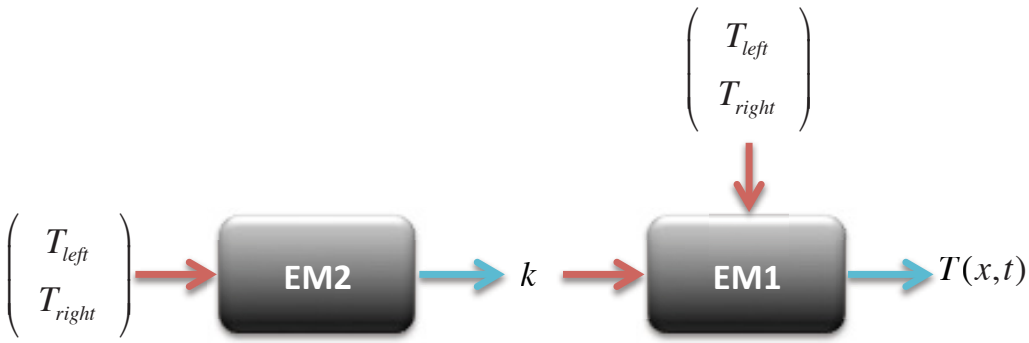


Figure 30. Heat conduction meta-model connections in a chain of evaluations.

2. In the second test the model *EM2* has as input the average temperature in the slab that is one of the outcomes of the model *EM1*. In addition, the model EM1 needs the output (thermal conductivity) of the model EM2 to compute the heat conduction problem. Therefore this scenario resolves in a non-linear system of equations. This is shown in Figure 31.

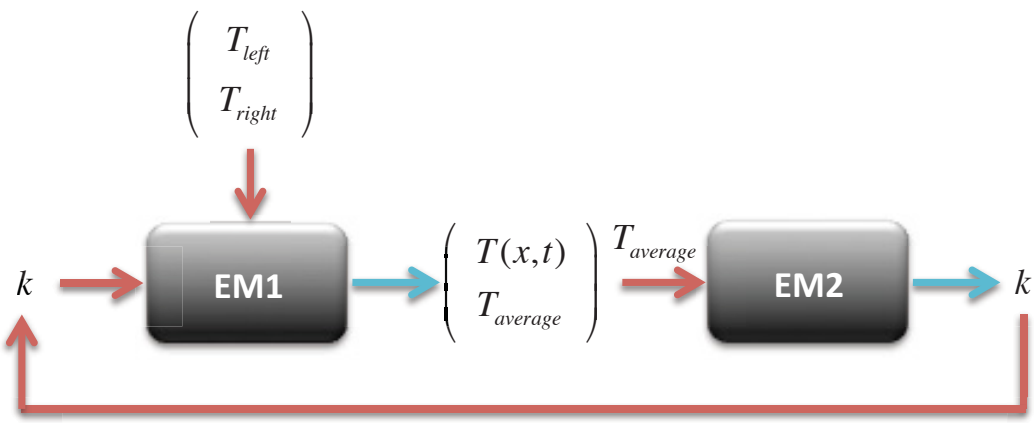


Figure 31. Heat conduction meta-model connections in a non-linear system.

Figure 32 and Figure 33 show the outcomes of the meta-model sampling in the two cases. As it can be seen, the obtained results match perfectly.

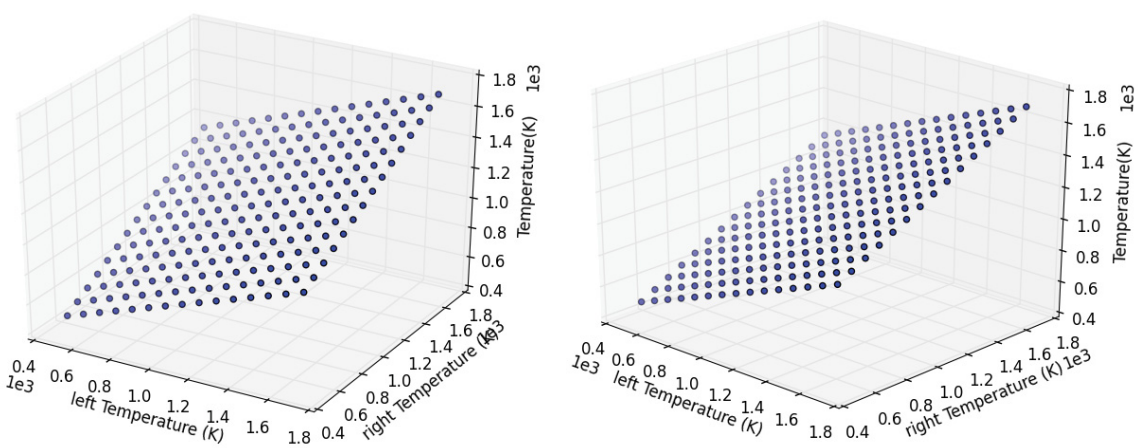


Figure 32. Temperature at mid-plane: sequential model (left) and Picard iteration (right).

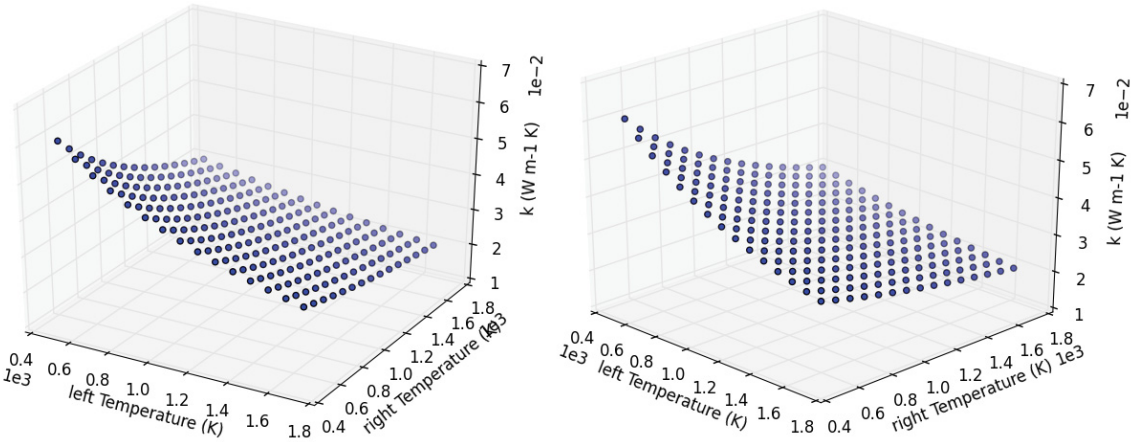


Figure 33. Compute thermal conductivity: sequential model (left) and Picard iteration (right).

4.5 Final Remarks

In this portion of the report, a newly developed capability of the RAVEN code has been shown. Through the meta-model entity, RAVEN is able to combine multiple models (i.e., simulation codes and ROMs), constructing a pipe network in order to transfer information among them. The addition of the Picard's iteration scheme lets the user solving combinations of models that resolve in non-linear systems.

5. CONCLUSION

This report highlights the following three major accomplishments:

1. Testing and improving of the reliability surface searching algorithm in RAVEN
2. Demonstration of the possibility to construct reduced order models for high fidelity multi-physics problems
3. Testing of an initial infrastructure, inside the RAVEN code, for the coupling of multiple surrogate models

Firstly the report explore the possibility to improve the greedy, reliability, surface-adaptive searching algorithm implemented in RAVEN by a lesser greedy, more robust searching algorithm. As expected, there is no a priori answer to this question; however, the performed tests are encouraging and less greedy algorithms have been tested and look promising. One of the option tested, the introduction of the topological analysis of the scoring function shows two benefits: one, it could be tuned to filter out artifacts present in the scoring function (e.g., tuning the persistence requirement for the scoring function extreme), and, two, in intrinsically stochastic systems, it allows a de-noising of the scoring function to avoid cases where the extreme is just a product of the statistical noise. In conjunction with the topological analysis of the scoring function, the introduction of a batching approach for selection of realization of the input space to be evaluated by the high-fidelity simulation has also been tested. This strategy is less greedy and, even though it might converge slower, it might represent a safer pattern with respect to the originally implemented approach in RAVEN. Currently, those options are undergoing RAVEN quality assurance approval and will be soon merged with the multi-grid acceleration[10].

Sections 3 and 4 laid the theoretical background and basic infrastructure for the next development period for the RAVEN code, which will allow even more challenging problems in reliability analysis to be addressed.

Section 3 describes the feasibility of surrogate model construction, following an ensemble approach for coupling of multi-physics models, where the exact coupling involves communication of what is

referred to as high-density fields (e.g., temperature and power). Those fields could be characterized by billions of degrees of freedom; therefore, they not suitable for being represented by surrogate models. In reality, the nature tendency to diffusion in physical system makes the existence of such a large number of uncorrelated degrees of freedom unlikely. Perturbations of the input space, rather than exciting each degree of freedom separately, tend to move the system response without altering its fundamental shape. For this reason, as shown in Section 3, perturbations of the input space led to variation in the high-density field (i.e., output) that could be represented by a very small set of directions where the system response changes. When multiple physics are connected to one another, the reduction of the active number of degree of freedom could become even larger because multiple physics act as a sequential set of filter applied to the initial perturbation in the input space.

This approach will make it possible to perform reliability analysis, starting from high-fidelity representation of complex systems. Those types of representations (e.g., RELAP-7 coupled to BISON and MAMMOTH) traditionally have a very expensive simulation cost that make them impossible because of thousands of simulations. Following the approach described in this report, it will be possible to create accurately-driven surrogate models for each single physics and coupling them a posteriori. This will allow a great computational saving and application of advanced reliability surface searching algorithms (similar to the one presented in Section 2).

This new approach has a large impact on reliability analysis and on uncertainty quantification, risk analysis, system optimization, and model calibration. In general this is true , for any application that requires a large number of simulations of the same complex system under perturbation of the input parameters. This very important result creates the possibility of constructing surrogates, within RAVEN, for a high-fidelity, multi-physics representation of complex systems.

Section 4 shows the early results for implementation of an ensemble approach for the coupling of surrogate models representing a multi-physics problem. While this is an initial implementation, the developed structure seems to support current needs and could be eventually extended in the future. This capability builds a complex system representation, even when the original models were not coupled, but just coupled their surrogate. Two applications are relevant for reliability analysis: (1) the possibility to build surrogate representation of a complex system, starting from libraries of surrogate models for each component, and (2) software implementation present during the first stage of surrogate model coupling when responses are high-density fields.

6. REFERENCES

1. Dan Maljovec, Bei Wang, Paul Rosen, Andrea Alfonsi, Giovanni Pastore, Cristian Rabiti, and Valerio Pascucci. Topology-inspired partition-based sensitivity analysis and visualization of nuclear simulations. Manuscript, 2015.
2. Klaus Brinker. Incorporating diversity in active learning with support vector machines. *Proceedings 20th International Conference on Machine Learning*, 2003.
3. David M W Powers. Evaluation: From precision, recall and F-Measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63,201.
4. Alkis Gotovos, Nathalie Casati, Gregory Hitz, and Andreas Krause. Active learning for level set estimation
5. Herbert Edelsbrunner, David Letscher, and Afra J. Zomorodian. Topological persistence and simplification.
6. Gunnar Carlsson, Afra J. Zomorodian, Anne Collins, and Leonidas J. Guibas. Persistence barcodes for shapes. *Proceedings Eurographs/ACM SIGGRAPH Symposium on Geometry Processing*, pages 124–135, 2004.

7. Samuel Gerber, Peer-Timo Bremer, Valerio Pascucci, and Ross T. Whitaker. Visual exploration of high dimensional scalar functions. *IEEE Trans. Vis. Comput. Graph.*, 16(6):1271–1280,2010.

8. Herbert Edelsbrunner and John Harer. Persistent homology - a survey. *Contemporary Mathematics*, 453:257– 282, 2008.

9. *Discrete and Computational Geometry*, 28:511–533, 2002.

10. Andrea Alfonsi, Cristian Rabiti, Diego Mandelli, Joshua Cogliati, Sonat Sen, and Curtis Smith. Improving limit surface search algorithms in raven using acceleration schemes. INL/EXT-15-36100, July 2015.

11. Brent Bryan, Jeff Schneider, Robert C. Nichol, Christopher J. Miller, Christopher R. Genovese, and Larry Wasserman. Active learning for identifying function threshold boundaries. *Advances in Neural Information Processing Systems*, 18,2005.

12. Brent Bryan. *Actively Learning Specific Function Properties with Applications to Statistical Inference*. PhD thesis, School of Computer Science, Dec 2007.

13. David Ginsbourger, Rodolphe Le Riche, Laurent Carraro, and De´partement Mi. A multi-points criterion for deterministic parallel global optimization based on gaussian processes. *Journal of Global Optimization*, 2009.

14. Robert Ghrist. Barcodes: The persistent topology of data. *Bullentin of the American Mathematical Society*, 45:61–75, 2008.

15. Yuhong Guo and Dale Schuurmans. Discriminative batch mode active learning. *Advances in Neural Infor- mation Processing Systems*, pages 593–600, 2008.

16. Dan Maljovec, Bei Wang, John Moeller, and Valerio Pascucci. Topology-based active learning. Technical Report Technical Report UUSCI-2014-00, SCI Insititute, University of Utah, 2014.

17. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

18. Dan Maljovec, Bei Wang, Ana Kupresanin, Gardar Johannesson, Valerio Pascucci, and Peer-Timo Bremer. Adaptive sampling with topological scores. *International Journal for Uncertainty Quantification*, 3(2):119– 141, 2013.

19. F. Gleicher, J. Ortensi, B. Spencer, Y. Wang, S. Novascone, J. Hales, D. Gaston, R. Williamson, and R. Martineau, "The Coupling of the Neutron Transport Application RATTLE-SNAKE to the Nuclear Fuels Performance Application BISON under the MOOSE Framework", International Topical Meeting on Advances in Reactor Physics, Kyoto, Japan, 2014.

20. G. Box, N. Draper, *Response Surfaces, Mixtures, and Ridge Analyses*, 2nd ed. Wiley: Hoboken, New Jersey, 1987.

21. N. Halko, P. Martinsson, and J. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions", *SIAM Review*, Volume 53, Number 2, pp.217-288, 2011.

22. Y. Bang, H. Abdel-Khalik, and J.M. Hite, "Hybrid Reduced Order Modeling Applied to Nonlinear Models", *International Journal for Numerical Methods in Engineering*, 91, Issue 9, pp.929–949, 2012.

23. M. Abdo and H. Abdel-Khalik, "Propagation of Error Bounds due to Active Subspace Reduction," Transactions of American Nuclear Society, 110, 2014.

24. M. Abdo and H. Abdel-Khalik, "Development of Multi-Level Reduced Order Modeling Methodology," Transactions of American Nuclear Society, San Antonio, Texas, 112, 2015.

25. H. Abdel-Khalik, Y. Bang, J. Hite, C. Kennedy, "*Reduced Order Modeling For Nonlinear Multi-Component Models*," International Journal for Uncertainty Quantification, **2**, 4, 341-461, 2012.

26. P. Holmes, J. Lumley, G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press: Cambridge Monographs on Mechanics, 1996.

27. C. Wang, J. Hite, and H. Abdel-Khalik, "Intersection Subspace Method for Uncertainty Quantification," Transactions of American Nuclear Society, 111, 2014.

28. B. Khuwaileh, J. Hite, H. Abdel-Khalik, "Subspace Methods for Multi-Physics Reduced Order Modeling in Nuclear Engineering Applications," Proceedings of PHYSOR 2014, Kyoto, Japan, 2014.

29. B. Khuwaileh and H. Abdel-Khalik, "Employing Non-Converged Iterates for Reduced Order Modeling," SIAM Annual Meeting, July, 2014.

30. M. Abdo, C. Wang, and H. Abdel-Khalik, "Probabilistic Error Bounds for Reduced Order Modeling," 7th International Conference on Modelling and Simulation in Nuclear Science and Engineering, Ottawa, October, 2015; and International Conference on Mathematics and Computation, Nashville, TN, April, 2015.

31. Mayer, C.D. (2000). Matrix Analysis, Philadelphia: SIAM

32. J. Hite and H. Abdel-Khalik, "Dimensionality Reduction in Nonlinear Optimization," Transactions of the ANS Winter Meeting, 2011.

33. D. Mandelli, C. Smith, A. Alfonsi, C. Rabiti, J. Cogliati, H. Zhao, I. Rinaldi, D. Maljovec, P.Talbot, B. Wang, V. Pascucci "Reduced Order Model Implementation in the Risk-Informed Safety Margin Characterization Toolkit." INL/EXT-15-36649 (September 2015)