

Generation and Use of Reduced Order Models for Safety Applications Using RAVEN

D. Mandelli[†], A. Alfonsi, C. Smith, C. Rabiti

Idaho National Laboratory, 2525 North Fremont Street, Idaho Falls (ID)

[†] Corresponding author: diego.mandelli@inl.gov

INTRODUCTION

The Risk-Informed Safety Margin Characterization (RISMC) [1] Pathway (as part of the Light Water Sustainability (LWRS) Program [2]) aims to develop simulation-based tools and methods to assess risks for existing Nuclear Power Plants (NPPs).

This Pathway, by developing new methods, is extending the Probabilistic Risk assessment (PRA) state-of-the-practice methods [3] which have been traditionally based on logic structures such as Event-Trees (ETs) and Fault-Trees (FTs) [4]. In more detail, the RISMC approach uses stochastic frameworks (i.e., RAVEN [5]) coupled with deterministic codes that model specific physical aspects of the plant (e.g., thermo-hydraulic and thermo-mechanic using RELAP5-3D [6] or RELAP-7 [7], and GRIZZLY [8] respectively).

One research direction is use of surrogate models, also known as Reduced Order Models (ROMs), as possible substitutes for one or more of the needed physical aspects. The use of ROMs can greatly reduce the computational cost of a single multi-physics simulation run. This advantage is relevant when many simulation runs need to be performed according to the desired stochastic analysis (usually through a stochastic sampling process).

RISMC APPROACH

A single simulator run can be represented as a single trajectory in the phase space. The evolution of such a trajectory in the phase space can be described as follows:

$$\frac{\partial \boldsymbol{\theta}(t)}{\partial t} = \mathcal{H}(\boldsymbol{\theta}, \mathbf{s}, t) \quad (1)$$

where:

- $\boldsymbol{\theta} = \boldsymbol{\theta}(t)$ represents the status of the system as a function of time t , i.e., $\boldsymbol{\theta}(t)$ represents a single simulation
- \mathcal{H} is the actual simulator code that describes how $\boldsymbol{\theta}$ evolves in time
- $\mathbf{s} = \mathbf{s}(t)$ represents the status of components and systems of the simulator (e.g., status of emergency core cooling system, AC system)

By using the RISMC approach, the PRA analysis is performed by:

1. Associating a probabilistic distribution function (pdf) to the set of parameters \mathbf{s} (e.g., timing of events)
2. Performing stochastic sampling of the pdfs defined in Step 1
3. Performing a simulation run given \mathbf{s} sampled in Step 2, i.e., solve Eq. (1)
4. Repeating Steps 2 and 3 M times and evaluating user defined stochastic parameters such CD probability (P_{CD}).

RAVEN FRAMEWORK

In order to perform PRA analyses of NPPs, the RISMC pathway is employing the RAVEN statistical framework, which is a recent add-on of the RAVEN package [9] that allows the user to perform generic statistical analysis. By statistical analysis we include: sampling of codes (e.g., Monte-Carlo [10] and Latin Hypercube Sampling [11], grid sampling, and Dynamic Event Tree [12]), generation of ROMs [13] (also known as surrogate models or emulators) and post-processing of the sampled data and generation of statistical parameters (e.g., mean, variance, covariance matrix).

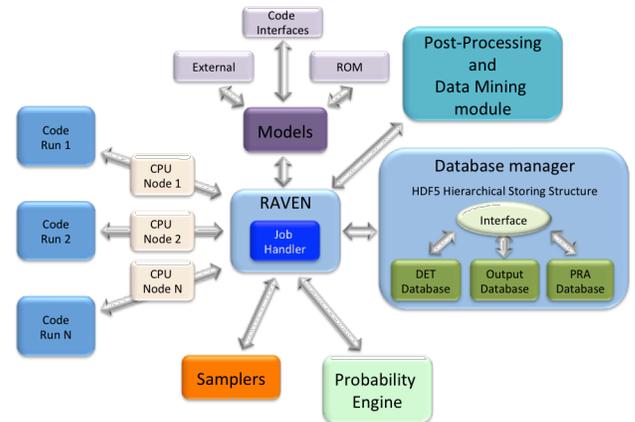


Fig. 1. Overview of the RAVEN statistical framework.

Figure 1 shows an overview of the elements that comprise the RAVEN statistical framework:

- Model: represents the pipeline between the input and output spaces. It is comprised of both interfaces for mechanistic codes (e.g., RELAP5-3D and RELAP-7) and ROMs

- Sampler: the driver for any specific sampling strategy (e.g., Monte-Carlo, Latin Hypercube Sampling, Dynamic Event Tree)
- Database: the data storing entity
- Post-processing: module that performs statistical analyses and visualizes results

RAVEN is interfaced with several codes and, actually, the users can build their own interfaces for the code they are interested in.

The interface for RELAP5-3D allows RAVEN to change specific values of any card contained in the RELAP5-3D input files according to the chosen sampling strategy.

In addition, at the end of each RELAP5-3D simulation run, RAVEN collects and stores all information generated from the output files (in the Database manager), it generates CSV files of the output data, and it processes such data through its internal Post-Processing and Data Mining module.

If multiple simulations need to be run, RAVEN has the capability to run simulations in parallel on multiple nodes and/or multiple CPUs. RAVEN applicability ranges from Linux based desktop/laptop to high performance computing machines.

As mentioned earlier, RAVEN has also the capability to “train” ROMs from any data set generated by any code. These ROMs are usually a blend of interpolation and regression algorithms and such “training process” basically consists of setting the optimal parameters of the interpolation and regression algorithms that best fit the input data set. Once the ROMs are generated, they can be used instead of the actual codes to perform any type of analysis since the generation of data from ROMs is much faster than the original code.

SURROGATE MODELS

A ROM is a mathematical model that aims to build a correlation given a set of data points. The starting point is typically a set of N data points:

$$(\mathbf{s}_i, \mathcal{H}(\mathbf{s}_i)) \quad i = 1, \dots, N \quad (2)$$

that samples the response of the original model. Given the set of these N data points, the ROM is trained and the resulting outcome is a model $\Theta(\mathbf{s})$ that approximates the original model $\mathcal{H}(\mathbf{s})$ (see Figure 2):

$$\Theta(\mathbf{s}): \mathbf{s}_i \rightarrow \Theta(\mathbf{s}_i) \cong \mathcal{H}(\mathbf{s}_i) \quad (3)$$

The advantage is the much faster computation of $\Theta(\mathbf{s})$ (e.g., RELAP) compared to the original model $\mathcal{H}(\mathbf{s})$. However, the evaluation of a ROM is affected by an intrinsic error which can not be bound and/or quantified.

We have identified two classes of ROM: model based and data based. These two classes are described in the next two sections.

Model Based ROMs

In model based ROMs the prediction is performed using a blend of interpolation and regression algorithms. Examples are:

- Gaussian Process Models (GPMs)
- Multi-dimensional spline interpolators

This class of algorithms has the advantage that they possess great prediction capabilities if the original $\mathcal{H}(\mathbf{s})$ is relatively smooth (i.e., no discontinuous).

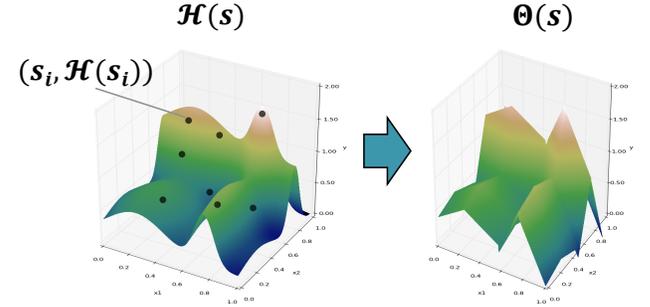


Fig. 2. Example of reduced order modeling approximation of a sampled 3-D response surface.

Data Based ROMs

In data based ROMs the prediction is performed by solely considering the input data by using data searching algorithms. Examples are:

- K nearest neighbor classifier (KNN)
- Graph based models

While the predictions of this class of ROMs is limited compared to model based ROMs, they have the advantage that they are able to handle very discontinuous $\mathcal{H}(\mathbf{s})$.

APPLICATIONS

For the scope of this article we have identified two possible applications of ROMs. These two applications are described in detail in the next two subsections.

Accelerator for stochastic analysis

For simulation-based safety applications, we aim to understand how a safety related parameter (e.g., maximum clad temperature) is affected by the timing and sequencing of events (e.g., recovery of AC power) or the uncertainties associated with characteristic parameters of the simulation. As an example, $\Theta(\mathbf{s})$ is used to reduce the number of samples in a Monte-Carlo analysis through adaptive sampling [14,15].

The scope is to determine the system failure probability by randomly sampling \mathbf{s} and simulating system behavior (e.g., maximum clad temperature). Failure probability p_F is calculated as the ratio of the number of simulations that lead to failure over the total number of simulations performed. Since p_F might be very small, a large number of computationally expensive simulations may be required.

Adaptive sampling infers, from a set of training simulations, regions that lead to failure (maximum clad temperature greater than failing temperature) and concentrates samples on those regions.

Note that the evaluation of $\Theta(\mathbf{s})$ for a new $\mathbf{s} \neq \mathbf{s}_i$ ($i = 1, \dots, N$) is much less computationally intensive than simulating the exact value $\mathcal{H}(\mathbf{s})$. This is in particular true when the evaluation of $\mathcal{H}(\mathbf{s})$ can take hours or days. Prediction capabilities lie within the ability to determine system outcomes (e.g., max clad temperature) in a much faster way than real-time.

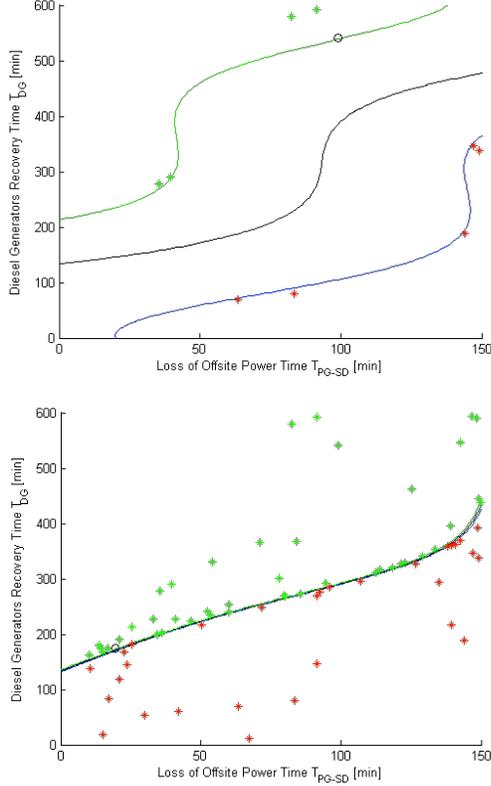


Fig. 3. Limit surface (black line) obtained for a simplified PWR system for a SBO scenario after 10 (top) and 60 (bottom) samples [14]. Uncertainty associated to the computed limit surface is indicated by the green and blue lines.

Prediction Models: Temporal emulators

In the previous section we introduced the concept of response surface methods and ROMs as tools to predict an approximated $\Theta(\mathbf{s})$ (which represents, for example, a simulated system response under an accident scenario) for a set of conditions specified in \mathbf{s} . The vector \mathbf{s} contains elements s_d such as timing and sequencing of events (e.g., recovery time of AC power, failure time of core cooling injection). Note that the value $\Theta(\mathbf{s})$ is a scalar and, thus, does not contain any temporal evolution type of information.

We extend the concept of ROM in order to be able to handle time dependent $\Theta(\mathbf{s})$: given \mathbf{s} , $\Theta(\mathbf{s}, t)$ is a time

dependent variable. In this case, the training consists of N points:

$$(\mathbf{s}_i, \mathcal{H}(\mathbf{s}, t)_i) \quad i = 1, \dots, N \quad (4)$$

Our approach is to start by dividing the temporal scale into intervals (assumed here to be of equal length but it is not required):

$$t = [t_1, \dots, t_T] \quad (5)$$

For each time point t_k ($k = 1, \dots, T$) we consider the subset of points:

$$(\mathbf{s}_i, \mathcal{H}(\mathbf{s}, t_k)_i) \quad i = 1, \dots, N \quad (6)$$

and we build the corresponding $\Theta(\mathbf{s})_k$. Thus, now we have a set of ROMs $\Theta(\mathbf{s})_k$ for each time point t_k ($k = 1, \dots, T$). The temporal predictor $\Psi(\mathbf{x}, t)$ is simply the vector of:

$$\Psi(\mathbf{x}, t) = [\Theta(\mathbf{s})_1, \dots, \Theta(\mathbf{s})_k, \dots, \Theta(\mathbf{s})_T] \quad (7)$$

In our applications, when each of the data points has been generated by safety analysis codes (e.g., RELAP, MELCOR [16]):

- \mathbf{s} is the configuration of the simulation (e.g., timing of events, values associated with uncertain parameters)
- $\Theta(\mathbf{s}, t)$ is the simulation associated with \mathbf{s} .

We performed a few tests with different types of datasets in order to identify performances and limitations of this algorithm. Figure 4 (top) shows a set of $n = 20$ simulations, i.e. $\mathcal{H}(\mathbf{s}, t)_i$ ($i = 1, \dots, 20$), generated by sampling two stochastic parameters, i.e. $\mathbf{s}_i = [s_1, s_2]$.

We initially divided the time scale uniformly $[0, 2500]$ into $T = 100$ intervals and for each time point t_k ($k = 1, \dots, 100$) we considered the data points $(\mathbf{x}_i, \mathcal{H}(\mathbf{s}, t_k)_i)$ ($i = 1, \dots, 20$) and built the ROMs $\Theta(\mathbf{s})_k$.

We then tested the temporal predictor:

$$\Psi(\mathbf{s}, t) = [\Theta(\mathbf{s})_1, \dots, \Theta(\mathbf{s})_{100}] \quad (8)$$

for several \mathbf{s}_j ($j \neq i$) and compared them with the simulated $\mathcal{H}(\mathbf{s}, t)$.

Figure 4 (bottom) shows the predicted scenario $\Psi(\mathbf{s}, t)$ (green line) and the actual simulated scenario $\mathcal{H}(\mathbf{s}, t)$. For this particular case we built $\Psi(\mathbf{s}, t)$ using Support Vector Machines [17] as basic ROM. A useful feature is that these algorithms are also capable of providing the uncertainty associated with the predicted results.

CONCLUSIONS

In this paper we have discussed possible applications of ROMs from a safety point of view. We have presented basic classes of ROMs that are available and their pros/cons. We have shown how it is possible to employ them to reduce the number of samples requires by a simulation based PRA (also know as dynamic PRA).

Lastly we have extended the concept of ROM to temporal ROMs that can emulate completely the temporal behavior of a generic system simulator code.

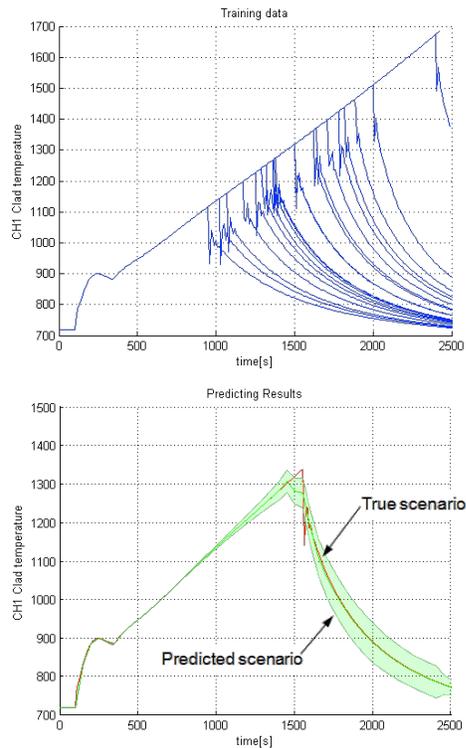


Fig. 4. Example of predicted temporal profile (red line in the bottom figure) given a set of simulated scenarios (top figure).

REFERENCES

1. C. SMITH, C. RABITI, AND R. MARTINEAU, "Risk Informed Safety Margins Characterization (RISMIC) Pathway Technical Program Plan", Idaho National Laboratory INL/EXT-11-22977 (2011).
2. DOE-NE Light Water Reactor Sustainability Program and EPRI Long-Term Operations Program – Joint Research and Development Plan, Revision 4, INL-EXT-12-24562, April 2015.
3. U.S. NRC, NUREG 1150, "Severe accident risks: an assessment for five U.S. nuclear power plants," Division of Systems Research, Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission, Washington, DC (1990).
4. M. ELISABETH PATE-CORNELL, "Fault Trees vs. Event Trees in Reliability Analysis", *Risk Analysis*, **4**, no. 3 (1984).
5. A. ALFONSI, C. RABITI, D. MANDELLI, J. COGLIATI, AND R. KINOSHITA, "Raven as a tool for dynamic probabilistic risk assessment: Software overview," in *Proceeding of M&C2013 International Topical Meeting on Mathematics and Computation*, CD-ROM, American Nuclear Society, LaGrange Park, IL (2013).
6. RELAP5 Code Development Team. RELAP5-3D Code Manual. INL, (2012).
7. A. DAVID, R. BERRY, D. GASTON, R. MARTINEAU, J. PETERSON, H. ZHANG, H. ZHAO, L. ZOU, "RELAP-7 Level 2 Milestone Report: Demonstration of a Steady State Single Phase PWR Simulation with RELAP-7," Idaho National Laboratory: INL/EXT-12-25924 (2012).
8. B. SPENCER, Y. ZHANG, P. CHAKRABORTY, S.B. BINER, M. BACKMAN, B. WIRTH, S. NOVASCONE, J. HALES, "Grizzly Year-End Progress Report", Idaho National Laboratory report: INL/EXT-13-30316, Revision 0 (2013).
9. C. RABITI, D. MANDELLI, A. ALFONSI, J. COGLIATI, AND B. KINOSHITA, "Mathematical framework for the analysis of dynamic stochastic systems with the raven code," in *Proceedings of International Conference of mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2013)*, Sun Valley (Idaho) (2013).
10. E. ZIO, M. MARSEGUERRA, J. DEVOOGHT, AND P. LABEAU, "A concept paper on dynamic reliability via Monte Carlo simulation," in *Mathematics and Computers in Simulation*, **47**, pp. 371-382, (1998).
11. J. C. HELTON AND F. J. DAVIS, "Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems," *Reliability Engineering & System Safety*, **81**-1 (2003).
12. A. AMENDOLA AND G. REINA, "Dylam-1, a software package for event sequence and consequence spectrum methodology," in EUR-924, CEC-JRC. ISPRA: Commission of the European Communities (1984).
13. H. S. ABDEL-KHALIK, Y. BANG, J. M. HITE, C. B. KENNEDY, C. WANG, "Reduced Order Modeling For Nonlinear Multi-Component Models," *International Journal on Uncertainty Quantification*, (2012).
14. D. MANDELLI AND C. SMITH, "Adaptive sampling using support vector machines," in *Proceeding of American Nuclear Society (ANS)*, San Diego (CA), **107**, pp. 736-738 (2012).
15. DAN MALJOVEC, BEI WANG, DIEGO MANDELLI, PEER-TIMO BREMER AND VALERIO PASCUCI, "Adaptive Sampling Algorithms for Probabilistic Risk Assessment of Nuclear Simulations," in *Proceeding of International Topical Meeting on Probabilistic Safety Assessment and Analysis (PSA)* (2013).
16. R. O. GAUNTT, MELCOR Computer Code Manual, Version 1.8.5, Vol. 2, Rev. 2. Sandia National Laboratories, NUREG/CR-6119.
17. C. J. C. BURGESS, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Min. Knowl. Discov.* **2**-2, pp. 121-167 (Jun. 1998).