2019 International High-Level Radioactive Waste Management Conference, Knoxville, TN

SAND2019-1917C

HIGH FIDELITY SURROGATE MODELING OF FUEL DISSOLUTION FOR PROBABILISTIC ASSESSMENT OF REPOSITORY PERFORMANCE

Paul E. Mariner*, Laura P. Swiler*, D. Thomas Seidl*, Bert J. Debusschere*, Jonathan Vo*, Jennifer M. Frederick*, and James L. Jerden**

**Sandia National Laboratories, P.O. Box 5800, Albuquerque, NM 87123-0747 USA, pmarine@sandia.gov*
***Argonne National Laboratory, Lemont, IL, USA*

Two surrogate models are under development to rapidly emulate the effects of the Fuel Matrix Degradation (FMD) model in GDSA Framework*. One is a polynomial regression surrogate with linear and quadratic fits, and the other is a k-Nearest Neighbors regressor (kNNr) method that operates on a lookup table. Direct coupling of the FMD model to* GDSA Framework *is too computationally expensive. Preliminary results indicate these surrogate models will enable* GDSA Framework *to rapidly simulate spent fuel dissolution for each individual breached spent fuel waste package in a probabilistic repository simulation. This capability will allow uncertainties in spent fuel dissolution to be propagated and sensitivities in FMD inputs to be quantified and ranked against other inputs.*

I. INTRODUCTION

High fidelity prediction of waste package and waste form degradation processes for thousands of waste packages in a probabilistic repository performance assessment calculation is expensive. With thousands of waste packages, thousands of time steps, and hundreds of realizations in a simulation, these process models may need to be called a billion times per simulation.

GDSA Framework is open source repository simulation software built around the massively-parallel multi-physics code PFLOTRAN.[1] GDSA stands for Geologic Disposal Safety Assessment. An important short-term goal of the development of *GDSA Framework* (pa.sandia.gov) is to perform probabilistic repository simulations to identify sources of uncertainty to help prioritize future R&D. To achieve this short-term goal with today's computer resources, developers must consider ways to include the effects of expensive process models in total system simulations.

One way to reduce computational expense is to develop response surface surrogate models that can rapidly emulate the mechanistic process models. An ideal response surface surrogate model runs orders of magnitude faster than its parent mechanistic model and provides outputs identical to those of the mechanistic model. In practice, the speed increase is easy to achieve. The challenge is achieving acceptable accuracy.

In 2018, a team of modelers and mathematicians at Sandia National Laboratories began exploring the potential value of developing surrogate models for the Fuel Matrix Degradation (FMD) model.[2] The FMD model has been coupled with PFLOTRAN,[3] but the coupled model runs too slowly for a set of probabilistic repository-scale simulations. The surrogate modeling work has examined polynomial regression, polynomial basis adaptation methods for dimensionality reduction, tabulation using tree-based lookup methods, and artificial neural networks. Two approaches were chosen for continued development, a polynomial regression surrogate model approach and a lookup table approach that involves an advanced nearest neighbor regression technique. Section II describes the FMD process model, and Section III presents the two surrogate models along with preliminary results.

II. FUEL DISSOLUTION PROCESS MODEL

The FMD model is a mechanistic spent fuel dissolution model coded in Matlab and developed at Argonne National Laboratory and Pacific Northwest National Laboratory. The model calculates spent fuel dissolution rates as a function of radiolysis, alteration layer growth, diffusion of reactants through the alteration layer, temperature, and interfacial corrosion potential.[4] During execution it employs a one-dimensional (1D) reactive transport model to simulate diffusion and chemical reactions across this layer over time. The 1D domain, depicted in Fig. 1, extends 0.05 m from the fuel surface to the bulk water. It is divided into as many as 100 cells with increasing length toward the bulk water boundary cell.

To couple the FMD model with PFLOTRAN, a "coupled" FMD model was coded in Fortran. At each time step, PFLOTRAN calls the coupled FMD model to obtain a new dissolution rate. Coupling required PFLOTRAN to keep track of the 1D chemical profiles across the domain from the previous time step. It also required relevant inputs from the main PFLOTRAN simulation, such as

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

temperature, time, and environmental concentrations in the boundary cell. Dose rate is calculated in the coupled FMD model from time and burnup. A full list of FMD model inputs and outputs available for surrogate modeling is presented in Table I.

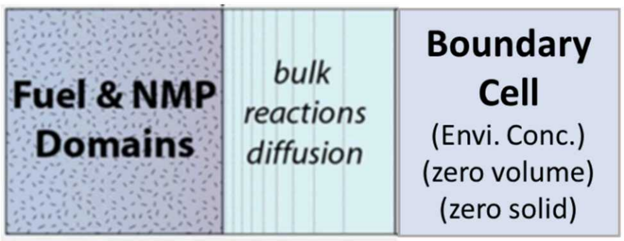

Fig. 1. FMD model domain.

TABLE I. Inputs/Outputs of Coupled FMD Model

Available Inputs	Outputs
• Initial concentration profiles across 1D corrosion/water layer (UO_2(s), UO_3(s), UO_4(s), H_2O_2, UO_2^{2+}, UCO_3^{2-}, UO_2, CO_3^{2-}, O_2, Fe^{2+}, and H_2) • Initial corrosion layer thickness • Dose rate at fuel surface (= f (time, burnup)) • Temperature • Time and time step length • Environmental concentrations (CO_3^{2-}, O_2, Fe^{2+}, and H_2)	• Final concentration profiles across 1D corrosion/water layer • Final corrosion layer thickness • Fuel dissolution rate

The coupled Fortran FMD model was tested on a problem involving a two-dimensional flow field containing 4 rows of 13 breached spent fuel waste packages. The model successfully simulated fuel dissolution for each of the waste packages over 100 time steps.[3] Of the 45 minutes of computational time required to run the simulation, 30 minutes were used calculating the fuel dissolution rates in the coupled FMD model.

III. SURROGATE MODELING

Two surrogate modeling approaches are presented, a polynomial regression surrogate model (Section III.A) and a k-nearest-neighbors surrogate model (Section III.B). The former provides a polynomial expression to emulate the FMD model while the latter uses an advanced technique to interpolate between points in a lookup table generated by the FMD model.

III.A. Polynomial Regression

It is often useful to construct a surrogate model to use in uncertainty and sensitivity analysis of a computational physics model when it is computationally demanding. A surrogate model (sometimes called meta-model, emulator, or response surface model) is an inexpensive input-to-output mapping that replaces a simulation code. Once constructed, this meta-model is relatively inexpensive to evaluate so it is often used as a surrogate for the physics model in uncertainty propagation, sensitivity analysis, or optimization problems that may require thousands to millions of function evaluations.[5]

There are many different types of surrogate models, including neural networks, regression models, radial basis functions, splines, etc. One popular approach in the literature is to develop an emulator that is a stationary smooth Gaussian process.[6,7] The popularity of Gaussian processes is due to their ability to model complicated functional forms and to provide an uncertainty estimate of their predicted response value at a new input point. There are many good overview articles that compare various meta-model strategies. Various smoothing predictors and nonparametric regression approaches are compared elsewhere.[5,7,8] Simpson et al. provide an excellent overview not just of various statistical meta-model methods but also approaches that use low-fidelity models as surrogates for high-fidelity models.[5] Haftka and his students developed an approach that uses ensembles of emulators or hybrid emulators.[9] Finally, polynomial chaos expansions (PCE) have become popular surrogate models over the past fifteen years.[10,11] These stochastic expansion methods approximate the functional dependence of the simulation response on uncertain model parameters by expansion in an orthogonal polynomial basis. The polynomials used are tailored to the characterization of the uncertain input variables.

III.A.1 Procedure

Our goal for the coupled FMD modeling effort is to develop a surrogate that can be called by PFLOTRAN as a replacement for the FMD model. Such a surrogate must be extremely fast to construct and evaluate, since it will be called repeatedly from PFLOTRAN for thousands of time steps and hundreds of waste packages. To start on this effort, we used a standalone MATLAB version of the FMD model to generate training data. The training data itself can be very large. For example, we may have hundreds of samples of FMD, where each sample involves a multi-dimensional vector sample of inputs such as the environmental concentrations, temperature, burnup, etc. The output is also extensive, since each FMD run involves hundreds of timesteps. So, a few hundred samples and a few hundred timesteps results in a large training matrix with tens of thousands of rows (each row being a training point at one particular timestep) and several columns of inputs (e.g., the left-hand quantities in Table 1) and one column of output (the fuel dissolution rate). Note that for this model, we are only interested in predicting the fuel dissolution rate although the other two output quantities could be treated with a surrogate in similar manner.

In our initial investigation, we decided to use polynomial regression surrogates for FMD, due to the large amount of training data, the smoothing characteristics of a regression model, and the requirement that the evaluation of the FMD surrogate be extremely fast. A linear regression model $\hat{f}$ as a function of an m-dimensional input vector $\boldsymbol{x} \in \mathfrak{R}^m$ is defined as:

$$\hat{f}(\boldsymbol{x}) \approx c_0 + \sum_{i=1}^{m} c_i\, x_i \qquad (1)$$

Similarly, a second order polynomial regression (also called a quadratic regression model) is defined as:

$$\hat{f}(\boldsymbol{x}) \approx c_0 + \sum_{i=1}^{m} c_i\, x_i + \sum_{i=1}^{m} \sum_{j\geq i}^{m} c_{ij}\, x_i x_j \qquad (2)$$

To determine the coefficients of the polynomial regression model, a least-squares formulation that minimizes the sum-of-squared error (SSE) between the surrogate model and the actual data is typically used.[12] The SSE is the standard error metric for overdetermined polynomial regression. It is a quadratic loss function which tends to find solutions near zero SSE well. We use the training data generated from the uncoupled Matlab FMD model in the SSE formulation. We have a matrix of n training samples, where each training sample has an input $\boldsymbol{x}_i$ and a corresponding output y_i. The coefficients minimize the SSE:

$$SSE = \sum_{i=1}^{n} \left(\hat{f}(\boldsymbol{x}_i) - y_i\right)^2 \qquad (3)$$

For general nonlinear regression problems, one needs to use optimization methods to find the vector of coefficients $\boldsymbol{c}$ which minimize the SSE. However, for linear regression models, the least squares problem reduces to a linear solve. If we write the entire sample matrix of inputs as $\boldsymbol{X}$ (of dimension n x m) and the sample matrix of outputs as $\mathbf{y}$ (of dimension n x 1), the optimization problem becomes:

$$\hat{\boldsymbol{c}} = \underset{\boldsymbol{c}}{argmin}\, \|\boldsymbol{X} \cdot \boldsymbol{c} - \boldsymbol{y}\|^2 = \boldsymbol{X}^{-1}\boldsymbol{y} \qquad (4)$$

In practice, we do not take the explicit inverse of the input sample matrix $\boldsymbol{X}^{-1}$ to solve for the optimal c but instead use a matrix factorization such as a QR factorization. This makes the determination of $\hat{\boldsymbol{c}}$ very efficient. Note also that this system is overdetermined for FMD: typically n = 100K or more but m (the number of coefficients) is on the order of 10 - 100.

A Latin hypercube sampling (LHS) study was performed to generate training and validation data for regression from the standalone Matlab FMD model. LHS is a stratified sampling technique that generates "well-spaced" samples; it typically gives lower variance statistical estimators than plain Monte Carlo sampling.[13] The six-dimensional sample space contained the parameters initial temperature, burnup, and the environmental concentrations of CO_3^{2-}, O_2, Fe^{2+}, and H_2. The probability distributions for each parameter are given in Table II.

TABLE II. LHS Sampling Input Parameters and Their Distributions

Parameter	Distribution	Min.	Max.
Init. Temp. (C)	Uniform	298	373
Burnup (Gwd/MTU)	Uniform	20	90
Env. CO_3^{2-} (mol/m^3)	Log-uniform	10^{-6}	10^{0}
Env. O_2 (mol/m^3)	Log-uniform	10^{-6}	10^{-1}
Env. Fe^{2+} (mol/m^3)	Log-uniform	10^{-6}	10^{-5}
Env. H_2 (mol/m^3)	Log-uniform	10^{-6}	10^{-1}

Nearly 5000 simulations were executed over the course of the LHS study. The temporal discretization in each problem consisted of 101 logarithmically-spaced (base 10) points from 0 to 10^5 years. In some simulations the UO_2 surface flux would superfluously stagnate after 10^4 years. We filtered the LHS results to remove any such runs and were left with data from approximately 3000 simulations. Our regression models were trained using data sets comprised of 1400 time series and tested on a set of 1400 different time series.

We built linear and quadratic surrogate models for the UO_2 surface flux (also referred to as fuel dissolution rate) using feature sets A and B. The members of feature set A are: time, temperature, environmental concentrations of CO_3^{2-}, O_2, Fe^{2+}, and H_2, dose rate at the fuel surface (all at the current timestep), and the previous timestep corrosion layer thickness and "interstitial" concentrations of UO_2^{2+}, $UO_2\,(CO_3)_2^{2-}$, UO_2, and H_2O_2 located at the bulk water boundary cell. For feature set B, the predictors from the current timestep are the same as in set A, but the sole predictor from the previous time step is the UO_2 surface flux. There is some hesitancy to use UO_2 surface flux from the previous time step (feature set B) because it does not mechanistically affect the flux for the current time step, neither in theory nor in the underlying FMD model. However, as the preliminary results show (next section), it is an excellent predictor for this model for the applied conditions. Note that because the fuel dissolution rate, time, and the environmental concentrations varied across orders of magnitude, we used the log-transformed values of these quantities in the regression model.

To assess the accuracy of the models for a specific training data size, we analyzed the relative pointwise absolute error (RPWAE). At each data point, this error is computed as:

$$RPWAE = \frac{|y_{pred} - y_{true}|}{y_{true}} = |1 - \frac{y_{pred}}{y_{true}}| \qquad (5)$$

For each training data set size, this error is averaged to obtain the mean RPWAE (M-RPWAE) metric for each test run.

III.A.2 Preliminary Results

Fig. 2 through 5 display 200 UO_2 surface flux traces from the test set (for visibility) and the corresponding predictions from the regression model. The results from the linear and quadratic models computed using feature set A are shown in Fig. 2 and 3. Fig. 4 and 5 present the results from feature set B. Table III contains the number of terms as well as the two test data validation metrics for each model: R-squared and mean absolute error values.

The quadratic model for feature set A is clearly better than the linear model. However, the use of higher-order regression models may result in overfitting.

We note that for feature set B the predictions are virtually identical between the linear and quadratic model. Thus, for simplicity and parsimony with comparable accuracy, we prefer the linear model.

TABLE III. Polynomial regression model summary for feature set, polynomial order, number of terms, test R-squared value, and test mean relative pointwise absolute error (M-RPWAE).

F-Set	P-Order	Terms	R^2	M-RPWAE
A	1	12	0.925	0.114
A	2	91	0.970	0.074
B	1	8	0.997	0.0256
B	2	45	0.997	0.0249

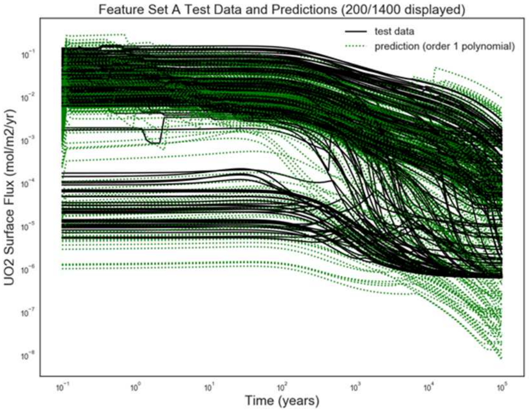

Fig. 2. Feature set A linear regression model.

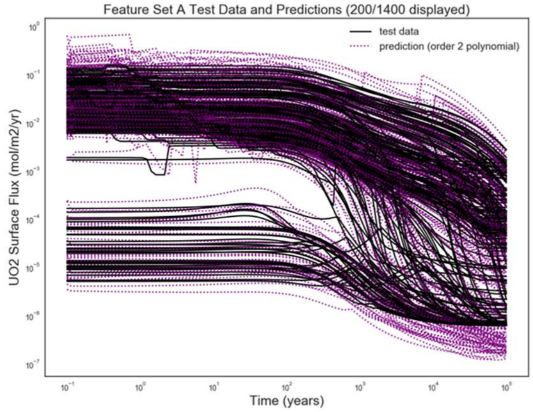

Fig. 3. Feature set A quadratic regression model.

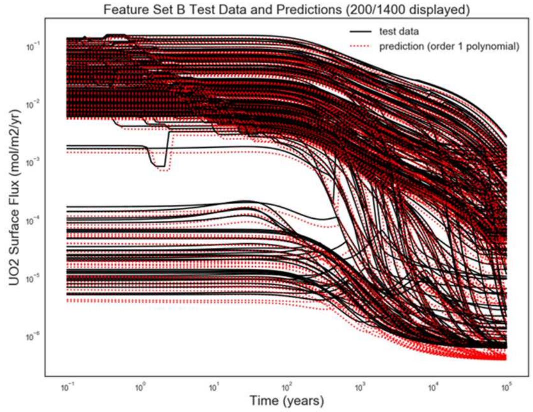

Fig. 4. Feature set B linear regression model.

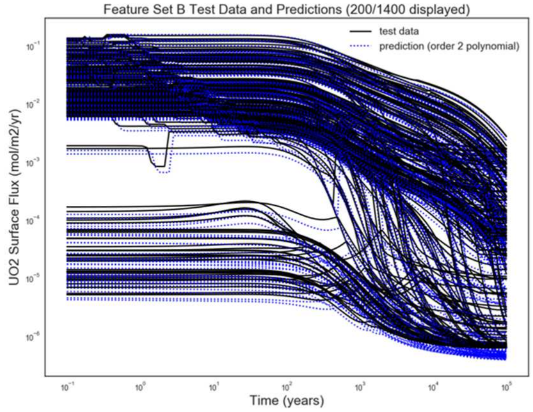

Fig. 5. Feature set B quadratic regression model.

III.A.3 Ongoing Work

At this point, we are in the process of incorporating the surrogate for FMD within PFLOTRAN, and we are also investigating slightly different regression formulations and feature sets. Once implemented in PFLOTRAN, we will be able to measure the performance and reduction in computational time relative to the coupled Fortran code.

III.B. k-Nearest Neighbor

The k-Nearest Neighbors regressor (kNNr)[14] is a supervised, non-parametric machine learning method that, unlike polynomial regression or neural networks, does not re-express the data in any way in order to make predictions. In contrast to the latter pair of methods, which are active learners, the k-Nearest Neighbors regressor is a lazy learner that tabulates data points inside of a domain X with labels Y to the end of using those values for predictions. This makes the kNNr highly interpretable, as no intermediate hypothesis selection process on the parameters is undertaken as with the aforementioned active learners. Instead, the label for a point within the domain but not in the "table" is obtained as an average of the labels of the k nearest neighbors of this new point, where $k \geq 1$ is fixed. The definition of nearest depends on the metric function one uses, though a typical choice is the Minkowski metric $(\sum_{i=1}^{d} |x_i - y_i|^p)^{\frac{1}{p}}$, with $p \geq 1$. The case of $p = 2$ is the popular Euclidean metric. The tabulation of data points can be implemented with a matrix representing entries in a table, however this is less efficient than modern tabulation methods like the K-D Tree or the Ball Tree.[15] The actual calculation of the predicted value need not be a uniform average. An inverse of the distance to each neighbor may be used to determine how influential that neighbor is in the final calculation of the weighted average.

One of the attractive features of kNNr is that it makes predictions based on local information only, and therefore does not require global smoothness over the input space. On the other hand, the approach requires a sufficiently dense table to get good predictive accuracy, and the cost of table look-ups increases as the table density increases.

III.B.1 Procedure

The kNNr is being considered as a surrogate model for predicting the UO_2 surface flux (also called UO_2 dissolution rate) in the waste package model component of PFLOTRAN. To that end, a sufficiently-dense table is generated based on samples from a MATLAB version of the original model. To improve numerical stability and to put all dimensions on similar footing despite the wide range of tabulated values, we take the log of all the entries of the table.

In this work, we utilized the kNNr method as implemented by scikit-learn, vz. 0.19.1.[15] This version of kNNr allows for several different kinds of distance metrics, including the Minkowski one. It also provides uniform and distance-based methods of weighting the average. Additionally, it allows for a few different methods of tabulation, one of which scales well with dimension: the BallTree tabulation method.

To assess the suitability of kNNr as a surrogate model, we analyzed the convergence of the kNNr accuracy of UO_2 surface flux predictions as a function of the amount of training data. The training data consists of time-traces of UO_2 surface flux obtained with the detailed FMD model for the sampled values shown in Table II. With 101 time points per trace and a number of runs M that is nearly 3000 runs, we have a total of about 3×10^5 data points. The 3000 runs are the same ones described in Section III.A.1 in the Polynomial Regression surrogate section, after Table II was introduced.

The number of runs from the dataset used for testing was set to 10% of the overall data set, with the remaining data available for training. In each experiment, 10 data sets of linearly increasing size were selected as training data sets to study the convergence as a function of the amount of training data. To account for randomness, an ensemble of 60 different permutations of the training data was generated for each training data set size.

For the results that will be discussed in this paper, we picked the Manhattan distance metric, or the Minkowski metric for $p = 1$, as it is better suited to higher-dimensional domain spaces, which is the same reason as to why the BallTree tabulation method was chosen. In the averaging, the distance-weighted approach was used. To predict the UO_2 surface flux, we used the following five features from the spatial mesh inside the FMD model bulk reaction diffusion space (Fig. 1):

- H_2 concentrations at the leftmost and rightmost endpoints of the spatial mesh inside the FMD model
- H_2O_2 concentrations at the leftmost and rightmost endpoints of the spatial mesh
- Dose rate at the leftmost endpoint of the spatial mesh

These features were selected as they showed a strong functional connection to the UO_2 surface flux, as determined from scatter plots between the flux and feature values.

As was done for the polynomial surrogate models, the accuracy of the kNNr was analyzed using the relative pointwise absolute error (RPWAE) and mean RPWAE (M-RPWAE) metrics. The ensemble average of the error over all 60 permutations of the test data sets of these M-RPWAE values is denoted as the ensemble average of the M-RPWAE (EAM-RPWAE).

The selection of the k value was done through a model selection experiment. Fig. 5 shows the average over the test data set of the relative error at each predicted point, for a set of k-values ranging from 1 to 52. The purple dashed line indicates where the k with minimal error for the different amounts of k considered, which is at $k = 7$.

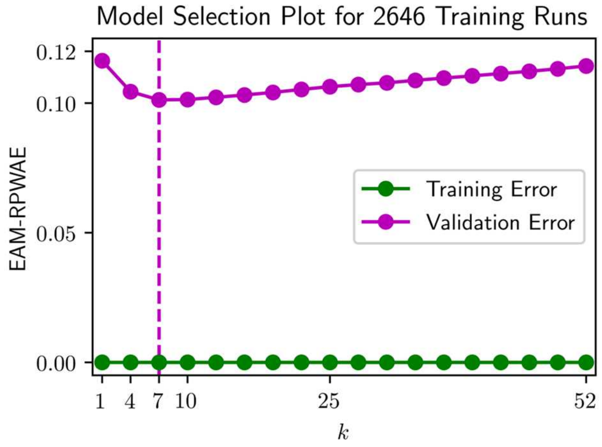

Fig. 5. Model selection results to determine the optimal number of nearest neighbors k.

III.B.2 Preliminary Results

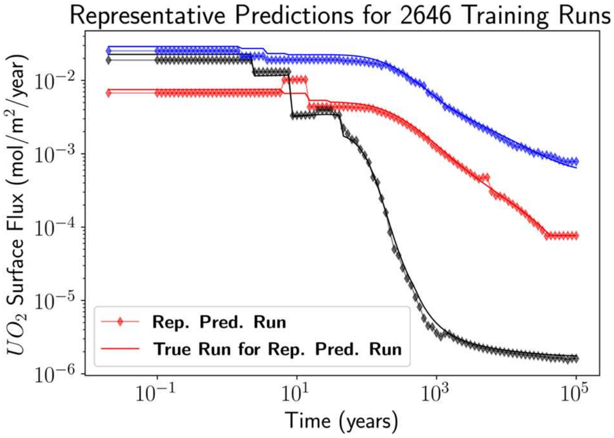

Fig. 6. The results of the trained kNNr on three members of the test set which are representative of the set with respect to the M-RPWAE.

Fig. 6 shows typical results of the kNNr with $k = 7$ for a table built on 2,646 training runs, where the lines with diamond symbols represent the approximation to the true time traces that are shown as lines without symbols of the same color. The axes of the plot are both log-scaled. Based on the plot, kNNr does rather well over the course of time.

As shown in Fig. 7, the average of the M-RPWAE decreases with increasing training data sizes, and the range of RPWAE averages per run also decreases, as indicated by the distance between the whiskers of the box plot. In this plot, the whiskers of the boxplot are the standard 1.5 * IQR (interquartile range), the orange line in each box represents the median, and outliers are not shown for the sake of readability. This shows the convergence of the kNNr regressor with increasing training data size. However, this convergence of the EAM-RPWAE is fairly slow. To get a better insight into the nature of the errors, we studied the histogram of the relative errors over all points.

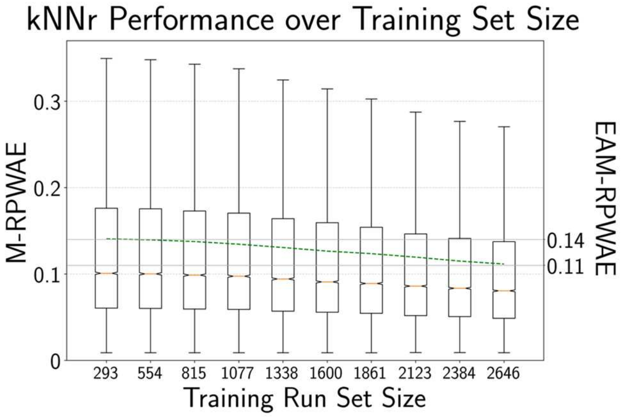

Fig. 7. Boxplots of M-RPWAE values as a function of training run set size.

Fig. 8 and 9 show histograms of the RPWAE values collected from all 60 random permutations of the training data sets for the case of 293 and 2646 training runs. This RPWAE distribution shrinks with increasing training size. Given the log-scale for the y-axis, it is clear that most RPWAE values are nearly zero, however there are outliers that affect the M-RPWAE values which are plotted in Fig. 7. The range of the outliers does shrink with increasing training data sizes.

To better understand the cause of the high errors, we investigate how the approximation error depends on the distances between the query point and its nearest neighbors. To this end, Fig. 10 shows a scatter plot of the RPWAE for each test point against the average distance to its 7 nearest neighbors, for the case of 2646 training runs. While most of the points show errors below 10%, they can increase to way above 100% when the average Manhattan distance to the nearest neighbors exceeds a threshold that sits somewhere between 0.4 and 0.6 (denoted by black hashes in Fig. 10). Introducing a distance cutoff in the table look-up with this particular average Manhattan distance as the threshold may improve the accuracy of the predictions and drive the RPWAE values of Fig. 10 even further down. Alternatively, this relationship between the prediction error and distance to the nearest neighbors in the table can be used to determine the optimal density of the table.

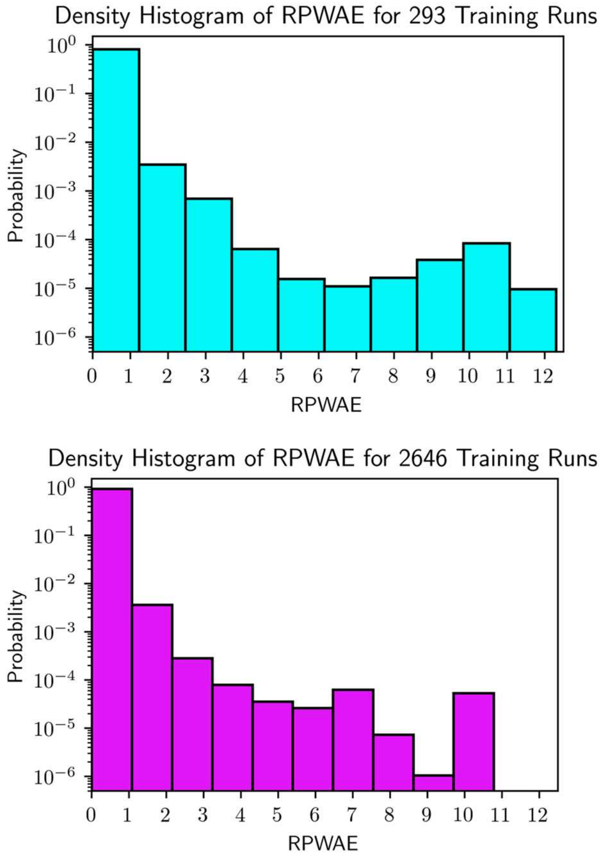

Fig. 8 and 9. Histogram density plots showing the probability distribution of RPWAE values for the smallest and largest training run set sizes used, respectively.

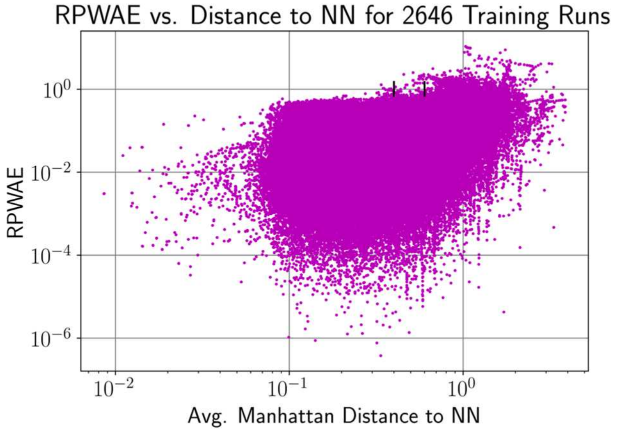

Fig. 10. Scatter plot of the RPWAE for each test point against the average distance to its 7 nearest neighbors, for the case of 2646 training runs.

III.B.3 Ongoing Work

To further improve the predictive fidelity of the kNNr, we are continuing to optimize the choice of features, as well as the amount of training data in the table. While adding more training data improves the accuracy of the predictions, a larger table requires more time to identify the nearest neighbors. Therefore, we will implement approaches to only add more data where it is needed to improve the accuracy. Further, in order to make this approach function as a standalone surrogate model for the full FMD model, we will build surrogate models for the evolution of the H_2 and H_2O_2 concentrations at the fuel surface and boundary cell and dose rate at the fuel surface as a function of the environmental conditions outside the fuel cask. These species concentrations are inputs for the surface flux surrogate model, but are currently integrated in time by the full FMD model.

IV. CONCLUSIONS

Two surrogate models are under development to rapidly emulate the effects of the Fuel Matrix Degradation (FMD) model in *GDSA Framework*. One is a polynomial regression surrogate with linear and quadratic fits, and the other is a k-Nearest Neighbors regressor (kNNr) method that operates on a lookup table. Preliminary results indicate that both approaches have a high degree of accuracy. However, more work is needed to refine the sample space, optimize the predictors for the sample space, and test the surrogate models in realistic repository PA simulations.

The aim of these surrogate models is to enable *GDSA Framework* to simulate spent fuel dissolution for each individual breached spent fuel waste package in a probabilistic repository simulation. Having the ability to emulate spent fuel dissolution in probabilistic PA simulations will have the added capability of allowing uncertainties in spent fuel dissolution to be propagated and sensitivities in FMD inputs to be quantified and ranked against other inputs.

ACKNOWLEDGMENTS

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. SAND2019-xxxx C.

REFERENCES

1. SNL. *GDSA Framework: A Geologic Disposal Safety Assessment Modeling Capability*. Available from: pa.sandia.gov (2017).
2. Jerden, J., J.M. Copple, K.E. Frey, and W. Ebert, *Mixed Potential Model for Used Fuel Dissolution - Fortran Code*, FCRD-UFD-2015-000159, US Department of Energy, Washington, DC (2015).
3. Mariner, P.E., W.P. Gardner, G.E. Hammond, S.D. Sevougian, and E.R. Stein, *Application of Generic Disposal System Models*, FCRD-UFD-2015-000126, SAND2015- 10037 R, Sandia National Laboratories, Albuquerque, New Mexico (2015).
4. Jerden, J., G. Hammond, J.M. Copple, T. Cruse, and W. Ebert, *Fuel Matrix Degradation Model: Integration with Performance Assessment and Canister Corrosion Model Development*, FCRD-UFD-2015- 000550, US Department of Energy, Washington, DC (2015).
5. Simpson, T.W., V. Toropov, V. Balabanov, and V. F.A.C. *Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come or not*. in *Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. 2008. Victoria, British Columbia, Canada. AIAA Paper 2008-5802.
6. Rasmussen, C.E. and C.K.I. Williams, *Gaussian Processes for Machine Learning*. MIT Press (2006).
7. Santner, T., B. Williams, and W. Notz, *The Design and Analysis of Computer Experiments*. New York, New York: Springer (2003).
8. Storlie, C.B., L.P. Swiler, J.C. Helton, and C.J. Sallaberry, "Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models," *Reliability Engineering & System Safety*, **94**(11), 1735-1763 (2009).
9. Viana, F.A.C., R.T. Haftka, and V. Steffen, "Multiple surrogates: how cross-validation errors can help us to obtain the best predictor," *Structural and Multidisciplinary Optimization*, **39**(4), 439-457 (2009).
10. Xiu, D., *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton University Press (2010).
11. Ghanem, R. and P. Spanos, *Stochastic Finite Elements: A Spectral Approach*. New York, New York: Springer Verlag (2002).
12. Seber, G.A.F. and C.J. Wild, *Nonlinear Regression*. New York, New York: Wiley & Sons (2003).
13. Helton, J.C. and F.J. Davis, "Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems," *Reliability Engineering & System Safety*, **81**(1), 23-69 (2003).
14. Ben-David, S. and S. Shalev-Shwartz, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, United Kingdom: Cambridge University Press (2014).
15. Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, **12**, 2825-2830 (2011).