Review Copy, 2019 International High-Level Radioactive Waste Management Conference, Knoxville, TN SAND2019-0641C

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] 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 50 μm 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

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.

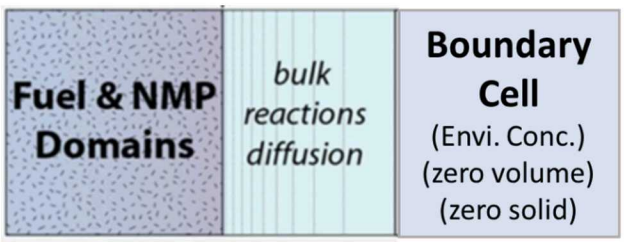

Fig. 1. FMD model domain

TABLE I. Inputs/outputs of a time-step-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 in this paper, a polynomial regression surrogate model (Section III.A) and a 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] 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.[7-9] 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.

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 outputs (the right-hand quantities in Table 1).

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 $\boldsymbol{x} \in \Re^m$ 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.[8] We use the training data generated from FMD 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:

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

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* = 10K or more but *m* (the number of coefficients) is on the order of 10.

Results from a linear and quadratic fit to FMD training data are shown in Figures 2 and 3. The inputs to these models are: time, temperature, environmental concentrations of CO_3^{2-}, O_2, Fe^{2+}, and H_2, the dose rate, and the previous timestep value of UO_2 surface flux. The output is UO_2 surface flux (also referred to as fuel dissolution rate). Note that because the fuel dissolution rate and the environmental concentrations varied across orders of magnitude, we used the log transformed values of these quantities in the regression model.

The results are virtually identical between the linear and quadratic model. This is not always the case, but here the linear model involved 8 terms and the quadratic model had 45 terms. The linear model had an R-squared value of 0.9890 and the quadratic model had an R-squared value of 0.9896. The mean absolute error values were 0.0987 for the linear model and 0.0982 for the quadratic model. Thus, for simplicity and parsimony with comparable accuracy, we choose the linear model.

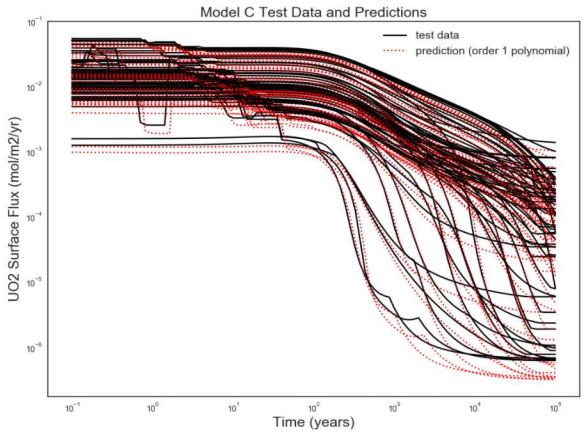

Fig. 2. Linear regression model

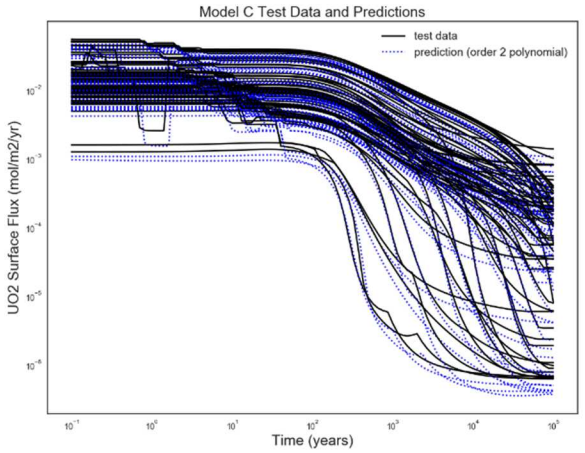

Fig. 3. Quadratic regression model

At this point, we are in the process of incorporating the surrogate for FMD within PFLOTRAN, and we are also investigating slightly different formulations and various input combinations.

III.B. k-Nearest Neighbor

The k-Nearest Neighbors regressor (kNNr)[5] 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, points within the domain but outside the "table" are based on the average of the labels of the k nearest neighbors of a new point in the domain that is not tabulated, 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 $\left(\sum_{i=1}^{d}|x_i - y_i|^p\right)^{\frac{1}{p}}$, 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.[6] Finally, the actual calculation of the predicted value need not be a uniform average. An inverse of the distance of 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.

The kNNr is a possible candidate for being a surrogate model to the waste package process model component of the PFLOTRAN reservoir model. To that end, a sufficiently-dense table will be generated based on samples from a MATLAB version of the original model. To improve numerical stability and to put all dimensions on the same footing, the entries of this table are rescaled along each dimension so that they fall between 0 and 1. Note that the points testing set are rescaled according to the means, maxima, and minima of the dimensions of the training set to prevent data snooping (giving the regressor any aspect of the answer in order to make predictions).

To use the kNNr, we utilized the method as implemented by scikit-learn, vz. 0.19.1. [6] This version of kNNr allows for several different kinds of distance metrics, including the Minkowski one, and uniform and distance-based methods of weighing 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 are currently evaluating the convergence of the kNNr accuracy of UO_2 surface flux predictions as a function of the amount of training data. 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. The arguments used include corrosion rate, temperature, environmental concentrations of CO_3^{2-}, O_2, Fe^{2+}, and H_2, and dose rate values at 40 positions along the spatial mesh. The output is UO_2 surface flux. Preliminary experiments were performed with k set to 1, 2, 3, and 5, while using the uniform weighting approach. Ultimately, setting k equal to 1 was the best choice for accuracy and efficiency.

For a given training set size, random permutations of the test set were performed, from which 20 test runs were selected at random. The RMS error of the predicted trajectories (using the kNNr with BallTree tabulation for the corresponding training set) is then computed for each test run and is normalized by the range of that test run. Fig. 4 shows a typical result of the kNNr applied to a random test set when using 190 training runs. Fig. 5 shows the resulting ensemble of normalized RMS errors as a function of the training data size. Furthermore, the normalized RMS errors were plotted against distances to the NN, which were collected via built-in methods in sk-learn's NearestNeighbors library. These distances were in turn normalized by the Manhattan norm of the test point; Fig. 6 contains the plot of normalized RMS error versus normalized distance to nearest neighbor. While the range of distance values carves out a distinct portion of the Cartesian plane, the fact that a given distance leads to multiple errors hints at the need to do some form of dimensionality reduction.

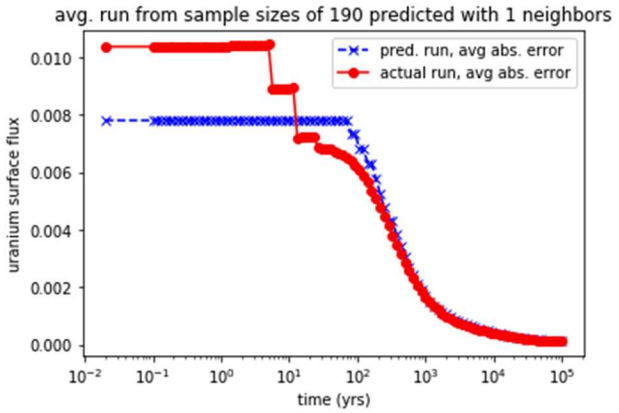

Fig. 4. kNNr applied to a random test set

Fig. 4 shows the results of the trained kNNr on a member of the test set which is representative of the set with respect to absolute pointwise error. One can see that the kNNr does very well in the later years, however there is some trouble fitting the earlier time points.

Fig. 5 shows a decreasing trend in the mean of the RMS error ensemble as sample size is increased (indicated by green dashed line), and the top end of the whisker generally goes down for increasing sample size.

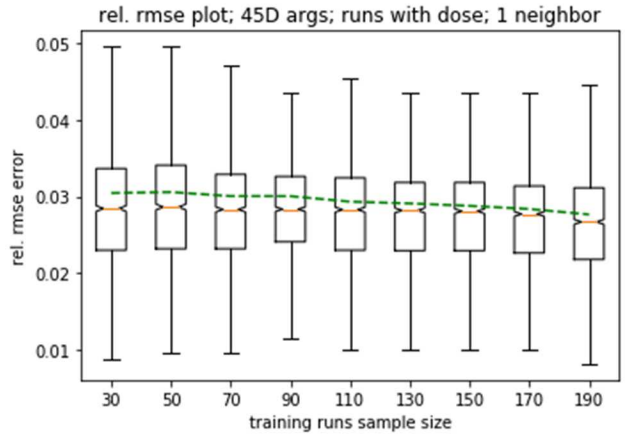

Fig. 5. Normalized RMS errors versus training data size for the kNNr method

Currently the results produced are from a limited data set. Additional data will be required to show that the kNNr method can achieve arbitrary accuracy with increasing training sample size. 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.

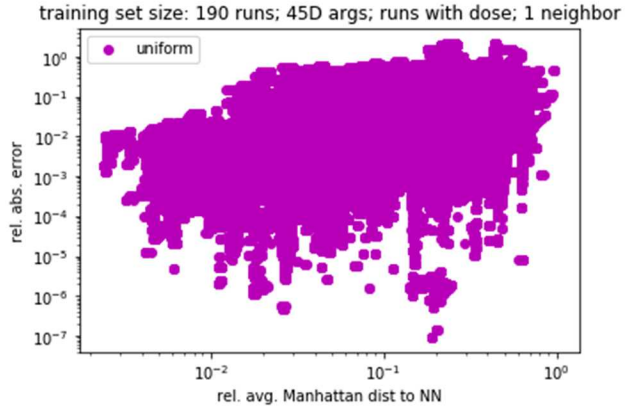

Fig. 6. Normalized RMS error versus normalized distance to nearest neighbor for the kNNr method

In Fig. 6, one can see that, though the overall average distances to the NN are not that big, that the error does not behave regularly as distance increases. The errors are relativized with respect to the Manhattan norm of the test point in the domain.

In order to investigate the convergence of the kNNr's predictive fidelity as a function of the amount of training data, additional training data is being gathered. On top of acquiring additional data, we are pursuing dimensionality reduction such as non-linear PCA to make the dependence of error on distance to NN in Fig. 6 less of a smear and closer to a functional dependence. Picking the right distance metrics will be an additional step for coarse-tuning the performance of the kNNr method.

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 these approaches are promising, but more work is needed to hone in on the optimal predictors and to extend the models to the necessary sample space. These surrogate models will enable *GDSA Framework* to simulate spent fuel dissolution for each individual breached spent fuel waste package in a probabilistic repository simulation. In addition, 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.

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-xxxxx

REFERENCES

1. SNL. *GDSA Framework: A Geologic Disposal Safety Assessment Modeling Capability*. 2017; Available from: pa.sandia.gov.
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. BEN-DAVID, S. AND S. SHALEV-SHWARTZ, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, United Kingdom: Cambridge University Press (2014).
6. 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).