Application of Gaussian Process Modeling to Analysis of Functional Unreliability

R. W. Youngblood[*]
Idaho National Laboratory, Idaho Falls, ID, USA

Abstract: Gaussian Process Models (GPMs) have been used widely in many ways [1]. The present application uses a GPM for emulation of a system simulation code. Such an emulator can be applied in several distinct ways, discussed below. Most applications illustrated in this paper have precedents in the literature; the present paper is an application of GPM technology to analysis of the functional unreliability of a passive containment cooling system, which was previously analyzed [2] using an artificial neural network (ANN), and later [3, 4] by a method called "Alternating Conditional Expectations" (ACE). The present exercise enables a comparison of both the processes and the results.

In this paper, (1) the original quantification of functional unreliability using ANN [2], and the later work using ACE [3], is reprised using GPM; (2) additional information provided by the GPM about uncertainty in the limit surface, generally unavailable in other representations, is discussed briefly; (3) a simple forensic exercise is performed, analogous to the inverse problem of code calibration, but with an accident management spin: given an observation about containment pressure, what can we say about the system variables?

Keywords: Gaussian Process, Simulation, Functional Unreliability

1. INTRODUCTION

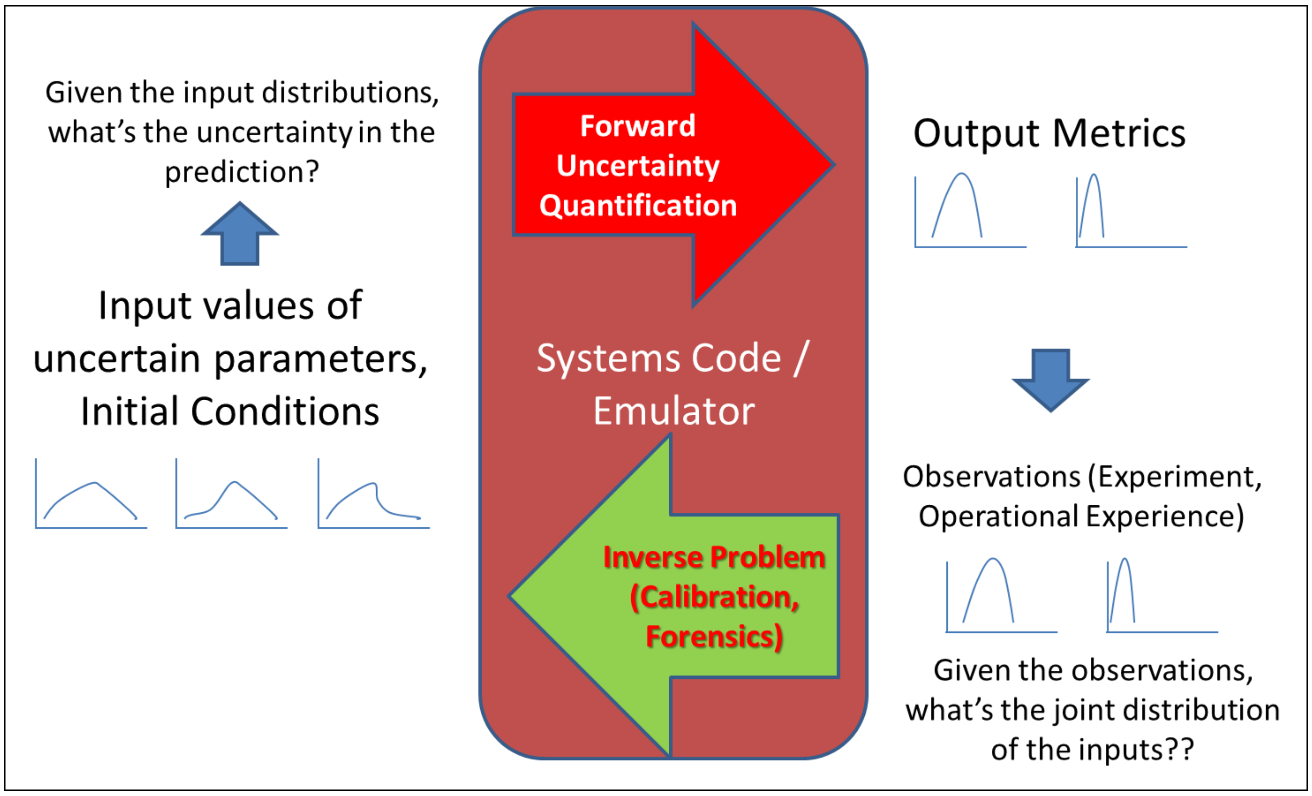
If a large number of complex system-level simulations are needed to support a decision – for example, if performance in a given input parameter space needs to be explored – and if it is impractical to run the simulation code itself a very large number of times – then fast-running surrogates are necessary. In the field of nuclear safety, surrogates have been used for decades in a variety of applications.

Figure 1 below compares more-traditional applications, such as forward uncertainty quantification, with more recent applications, such as code calibration. In forward uncertainty quantification, one propagates input uncertainties through a model to get information about the output uncertainty. If the inputs are sampled appropriately, the pdf of the output variables can then be applied for the purpose at hand, recognizing that uncertainty in the fidelity of the model is, of course, not addressed in such a simple process. If the model is too resource-intensive to be run as many times as are needed for the purpose at hand, one may be able to do enough runs to construct a surrogate, and then run the surrogate a large number of times. Some early work of this kind used so-called "response surfaces" as surrogates for thermal-hydraulic simulation codes, but concerns have been raised about the smoothness assumption embedded in simple response surface formulations. Later work has used other surrogates, such as ANNs, ACE, and GPMs.

As computer power increases and software improves, it is becoming possible to carry out some forward-propagation exercises without surrogates. However, there is currently significant interest in the inverse problem, which (at least in some implementations) is much more computationally intensive than the forward problem. For example, Markov-chain Monte Carlo (MCMC) is a widely known and widely practiced method for obtaining a joint pdf of "input" variables in a complex modeling situation, given observations that can be compared with model "outputs," and this method can correspondingly be used for calibration of simulation codes (i.e., reduction of epistemic uncertainty in their inputs). The efficiency of MCMC varies greatly depending on the specific implementation, but in general, MCMC requires many, many model evaluations, which, for complicated problems, will typically need to be done using surrogates, for the reasonably foreseeable future. GPMs are logical candidates for such surrogates, and their use in this way has been, and is being, undertaken widely.

[*] Robert.youngblood@inl.gov

Figure 1. Forward and Inverse Problems in Systems Analysis


Given the input distributions, what's the uncertainty in the prediction?
Input values of uncertain parameters, Initial Conditions
Forward Uncertainty Quantification
Systems Code / Emulator
Inverse Problem (Calibration, Forensics)
Output Metrics
Observations (Experiment, Operational Experience)
Given the observations, what's the joint distribution of the inputs??

Using a simple GPM, this paper reprises an early exercise in quantification of functional unreliability of a "passive" system [2], which was performed originally using an ANN, and later redone using a method called Alternating Conditional Expectations (ACE) [3, 4]. The capability of the GPM to say something about uncertainty in its predictions will be discussed briefly; and finally, the capability to execute the inverse problem will be explored briefly. As will be seen later, the original data set does not lend itself to a typical real application of the inverse solution capability, although the general approach of driving a GPM within MCMC seems to work well.

2. THE PASSIVE CONTAINMENT COOLING SYSTEM

The problem reanalyzed here involves the functional unreliability of the passive containment cooling system (PCCS) in the "Simplified Boiling Water Reactor (SBWR)," a "passive" design that was undergoing review at the USNRC in the early 1990's. A full description of this system is well beyond the scope of this paper, but the system is described in [2, 4]. Readers who are generally familiar with BWR technology may benefit from the following summary. The purpose of the PCCS is to remove heat from containment, specifically the drywell atmosphere. This is a broadly useful capability in the SBWR, because the SBWR response to many initiating events culminates in the RCS steaming directly to the drywell, with core makeup from a gravity-driven cooling system (GDCS). The PCCS is loosely analogous to an isolation condenser (IC), but whereas the IC condenses steam directly from the core by rejecting heat to an external pool of water, and returns condensate to the core, the PCCS condenses steam from the drywell atmosphere and returns condensate to the GDCS. Like the IC, the PCCS requires an adequate heat sink in the form of the external water pool, but it is passive.

However, the drywell atmosphere is not pure steam; it contains non-condensables. In the process of condensing steam from the drywell atmosphere, the PCCS may accumulate non-condensables, which would substantially reduce its functional efficiency if not for a design feature intended to push the non-condensables into the wetwell. If the efficacy of the PCCS is reduced by accumulation of non-condensables in a region where steam needs to migrate to a condensing surface, the drywell pressure should increase relative to the wetwell pressure, and this pressure difference is used to push the non-condensables into the wetwell, restoring the efficiency of the PCCS. Because this feature relies on a relatively small pressure difference between the wetwell and the drywell, it would be compromised by a leak between the two airspaces that allowed the pressure to equalize. This was originally accounted for in a fault-tree model as a failure of a vacuum breaker to seal. Of course, a really large leak between these airspaces could fail the pressure suppression function during blowdown, so the present work is focused on smaller leaks causing problems only after blowdown.

Questions have been raised by many workers for many years about the appropriateness of discrete-logic models for this kind of problem. Many passive systems rely on small pressure differences to drive circulation (e.g., due to gravity) in order to function, and influences that would be negligible in classical pumped systems (e.g., pressure drops that are small relative to the driving pressure) could therefore, in principle, affect the functioning of "passive" systems. Quantifying the functional unreliability of such systems therefore entails quantifying the probability of such influences being present, addressing continuously varying magnitudes of those influences, and properly accounting for synergistic effects of multiple such influences.

For the PCCS problem, this was undertaken in [2]; an accessible description of the work is provided in [4]. A simple CONTAIN model of the system was developed. Five variables were selected to define the issue space:

Table 1: The Issue Space

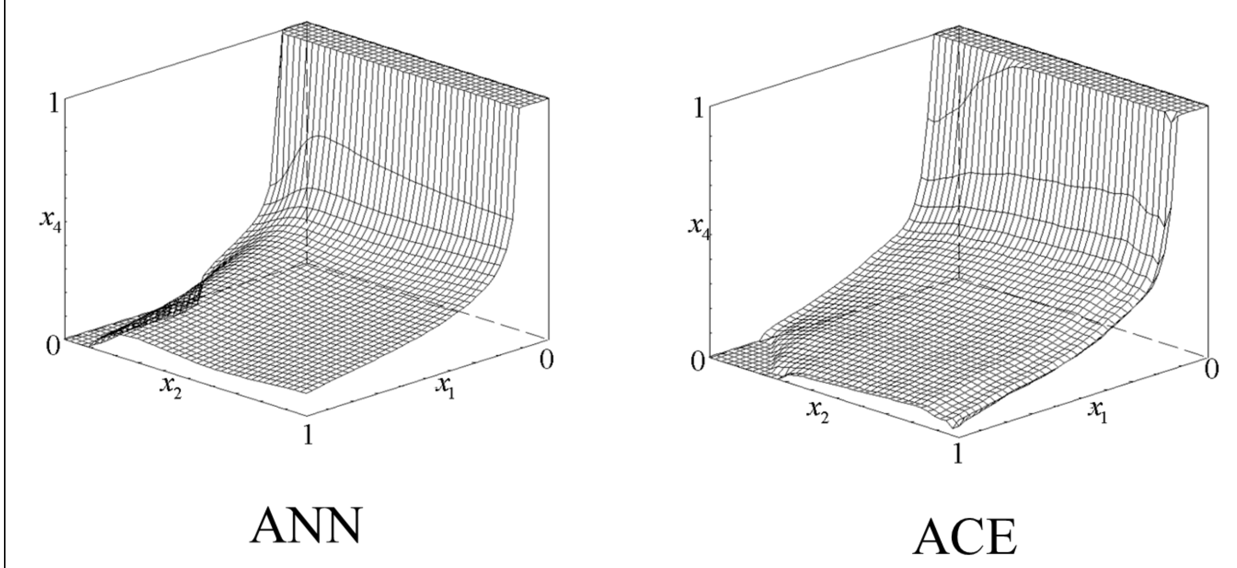
Variable	Meaning	Effect
X1	Area of leak between airspaces	Significant leak compromises PCC function
X2	PCCS Heat exchanger tube fouling	As modeled, increasing X2 increases heat transfer
X3	Flow loss coefficient of PCCS inlet line	Taken to extremes, could cut off PCCS flow, but within the range modeled, has little effect
X4	Flow resistance of GDCS line	Directly affects quench of the RCS following GDCS initiation; indirectly affects redistribution of non-condensables in the drywell
X5	Fraction of allowed reverse flow in GDCS that would be induced by back-pressure, due to postulated check valve failure	Postulated but not seen to occur in the simulations

In that work, which was focused on quantifying the functional unreliability, the emphasis was on identifying the "limit surface" (the surface in key-variable space that separates "success" from "failure", in this case overpressure). Note that the "limit surface" is different from the "response surface." The limit surface is somewhat analogous to the minimal cut sets for functional failure in a discrete-variable model, and has analogous uses: (1) given the limit surface, it is straightforward to quantify functional unreliability for any given distribution of the underlying variables, and moreover (2) the limit surface provides qualitative insight to an engineering discussion of what can cause functional failure. The quantification of unreliability in [2] depended on sampling the key variables, and determining whether each sample led to exceedance of the failure threshold in containment pressure. This exercise requires accurate classification of each sample into "success" or "failure," which requires accurate knowledge of the limit surface, but does not require high accuracy in the amount by which the threshold is succeeded or satisfied. A more modern treatment of the failure scenarios would, in fact, consider uncertainty / variability in the failure threshold, but such an extension was beyond the scope of this exercise.

In [2], simulation (CONTAIN) runs were chosen to optimize coverage of the region near the limit surface. This was done iteratively; some manually-chosen CONTAIN runs were performed to initialize an artificial neural network (ANN) representation of the limit surface, but thereafter, runs were chosen using a genetic algorithm whose objective function rewarded candidate CONTAIN runs that were assessed by the current ANN limit surface representation to be near the limit surface, and also far away from existing runs. This approach adequately concentrated runs near the limit surface. The result obtained after some 130 runs is shown below. Nowadays, this does not sound like very many runs, but at the time (early 1990's), this work was very time-consuming given the resources available to the project.

Figure 2. Visualization of the Limit Surface

ANN and ACE Limit Surface Projections



The left portion of Figure 2 shows a projection of the limit surface as represented by the ANN, with two of the five variables (X3 and X5) fixed for ease of visualization. The three variables shown are each scaled from 0 to 1 to cover the range assessed in [2]. It turns out that the limit surface varies relatively little as a function of X3 and X5, so this projection is an adequate representation for some purposes, even though the weak dependence on the other two variables is not shown explicitly.

The limit surface itself corresponds to pressure = .483 MPa; the region under the limit surface corresponds to success (peak pressure < .483 MPa), and the region above the limit surface corresponds to failure (peak pressure > .483 MPa). To understand the figure, begin at the origin (0, 0, 0). For X1=0, we do not see overpressure. As X1 (the leak area) increases with X2 and X4 fixed at 0, we eventually cross the limit surface, as expected. If we now allow X4 to increase, which increases GDCS flow and therefore RCS quench, and which consequently redistributes non-condensables in the drywell, we cross the limit surface at lower values of X1. Finally, the effects of X1 and X4 are partially offset if we allow X2 (heat transfer) to increase.

The right portion of Figure 2 shows a different representation of the same CONTAIN results, based on a method called "Alternating Conditional Expectations" (ACE), obtained later by [3]. Apart from some difference in the lower left corner, the two are broadly consistent, and, as presented below, yield generally similar results for functional unreliability. ACE has some advantages for this application, as discussed in [3, 4].

3. GAUSSIAN PROCESS MODEL

In the present exercise, the results described above are analyzed a third time, now using a simple Gaussian Process Model. The algorithms implemented in this exercise are all described in [1]. The covariance function implemented here is the squared-exponential, using a different length scale for each of the Xi (cf. section 5.1 of [1]). These length scales were “learned” from the data through MCMC, using a “Leave-One-Out” likelihood function (cf. section 5.4.2 of [1]). That process yielded the following estimates for the length scales:

Table 2: Length Scales Derived from Training Data

Variable	Length Scale
X1	2.50E-01
X2	1.10E+01
X3	3.18E+00
X4	6.09E-01
X5	3.59E+00

A {short, long} length scale in a given dimension allows for {rapid, slow} changes in the function value (pressure as a function of leakage, etc.), so the table of length scales is almost a kind of “sensitivity importance” ranking. According to the above table, the importance of X1 (magnitude of leakage) is, correspondingly, highest, as one would expect from the preceding discussion, followed by X4. Based on the discussion in [2], one would expect X2 to be more “important” than X3 or X5, which the table suggests is not the case. However, all three of these variables are relatively unimportant; since the {Xi} here are all scaled between 0 and 1, the length scales in X2, X3, and X5 over which output variation is significant are 3 to 10 times larger than the entire range of variation explored in this exercise. Therefore, it is difficult to refine them accurately without a lot of data, preferably over a broader range of variable values.

Given the GPM, which encodes the CONTAIN prediction of pressure as a function of {Xi}, it is straightforward to quantify functional unreliability for any desired set of input probability density distributions. [2, 3] considered 16 sets of pdfs: a “nominal” case (#1) (“Nom” in the table cells) reflecting nominal performance in the variables {Xi}, and 15 cases corresponding to various postulated combinations of degraded performance in these variables, modeled by shifting their pdf’s to “worse” cases. The table below compares the ANN results for functional unreliability obtained originally by [2] with the ACE results obtained later in [3], and, in the far right column, with the GPM-based results obtained here. For most cases, the agreement is reasonable. The GPM result seems generally a bit closer to the ANN result. These three sets of results are based on different analyses of the same underlying data set, and it is no longer practical to derive “truth” for this exercise (it would involve reconstituting a 20-year-old CONTAIN model, and running it many times); but given the diversity of analysis methods, it is noteworthy that the results are generally consistent.

Table 3: Comparison of Results for Functional Unreliability

Case	x_1	x_2	x_3	x_4	x_5	***P*** **(ANN)**	***P*** **(ACE)**	***P*** **(GPM)**
1	Nom	Nom	Nom	Nom	Nom	4.934×10^{-4}	3.418×10^{-4}	4.70E-04
2	5%	Nom	Nom	Nom	Nom	5.694×10^{-2}	4.746×10^{-2}	5.80E-02
3	10%	Nom	Nom	Nom	Nom	1.097×10^{-1}	0.952×10^{-1}	1.08E-01
4	Nom	5%	Nom	Nom	Nom	5.210×10^{-4}	4.005×10^{-4}	8.40E-04
5	Nom	10%	Nom	Nom	Nom	5.164×10^{-4}	4.081×10^{-4}	7.60E-04
6	Nom	Nom	5%	Nom	Nom	4.801×10^{-4}	2.669×10^{-4}	5.80E-04
7	Nom	Nom	10%	Nom	Nom	4.964×10^{-4}	2.557×10^{-4}	5.40E-04
8	Nom	Nom	Nom	5%	Nom	2.955×10^{-4}	1.863×10^{-4}	2.60E-04
9	Nom	Nom	Nom	10%	Nom	2.392×10^{-4}	1.460×10^{-4}	1.90E-04
10	Nom	Nom	Nom	Nom	5%	4.848×10^{-4}	3.515×10^{-4}	5.50E-04
11	Nom	Nom	Nom	Nom	10%	4.813×10^{-4}	3.510×10^{-4}	6.70E-04
12	5%	Nom	Nom	5%	Nom	3.964×10^{-2}	3.086×10^{-2}	3.97E-02
13	5%	Nom	Nom	10%	Nom	3.292×10^{-2}	2.515×10^{-2}	3.42E-02
14	10%	Nom	Nom	5%	Nom	7.915×10^{-2}	6.429×10^{-2}	8.08E-02
15	10%	Nom	Nom	10%	Nom	6.663×10^{-2}	5.303×10^{-2}	6.86E-02
16	5%	5%	5%	5%	5%	4.039×10^{-2}	2.980×10^{-2}	4.52E-02

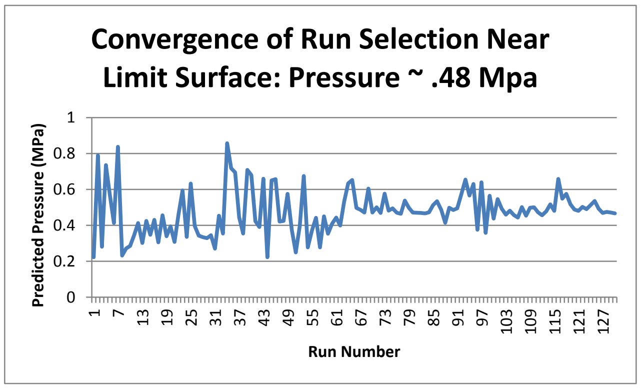
4. PREDICTIVE VARIANCE

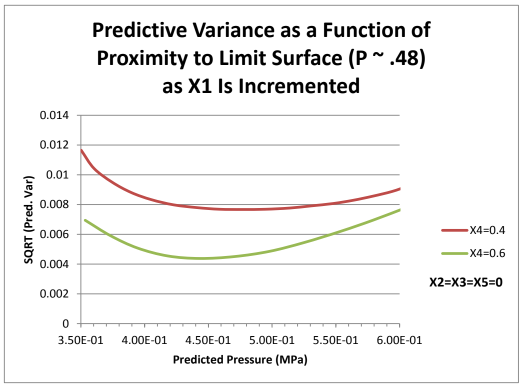
An interesting feature of GPMs is that they yield not only a prediction (a mean value) but also an associated "predictive variance." The GPM itself does not, of course, "know" about errors or modeling uncertainties in CONTAIN. In principle, we could have provided synthetic uncertainty information for each data point, but, in this exercise, have not done so. In the present exercise, then, predictive uncertainty in each point does not reflect a complete uncertainty assessment, but simply reflects how near the subject point is to the points in the training set, and, to some extent, how the GPM's covariance function was modeled and subsequently "trained" by the data.

Given that care was taken to concentrate the original CONTAIN runs near the limit surface, while also trying to promote coverage of the limit surface, we expect the variance to be generally smaller near the limit surface (or at least not larger) than it is far from the limit surface (except perhaps where a point being predicted happens to lie close to a training point that happens to be far from the surface). This appears to be the case. The leftmost portion of the figure below shows the original runs generated by the query learning algorithm, slowly converging on the limit surface (as evidenced by the increasing tendency to nail the limit pressure, 0.48), while the rightmost portion plots the square root of the predictive variance against the predicted pressure for two scans of X1 through the limit surface at different values of X4.

While the leftmost plot suggests that some convergence has occurred, it also suggests that given time, the results would benefit from additional code runs. *Of course*, more runs would be better; but beyond that, one sees in the left-hand portion of the figure instances where the predicted pressure exceeds the target by a significant margin.

Figure 3: Illustration of Predictive Variance





There has been little doubt for many years that GPMs are useful surrogates. It appears from the present discussion that the capability of GPMs to address predictive uncertainty explicitly could be harnessed to target additional code runs in a completely different way than was done in [2]. In [2], based on the perceived significance of the limit surface, and the need to classify samples according to the side of the limit surface on which they fell, priority was given to code runs that were near the surface but far away from previously performed runs. In light of the above, it would be natural to give priority to code runs in regions of the issue space that are significant but subject to large predictive variance, given the current information base. The notion of "significance" could be generalized, for example to include not only proximity to the limit surface but also probabilistic significance, based on the current probability density functions of the inputs.

5. FORENSICS

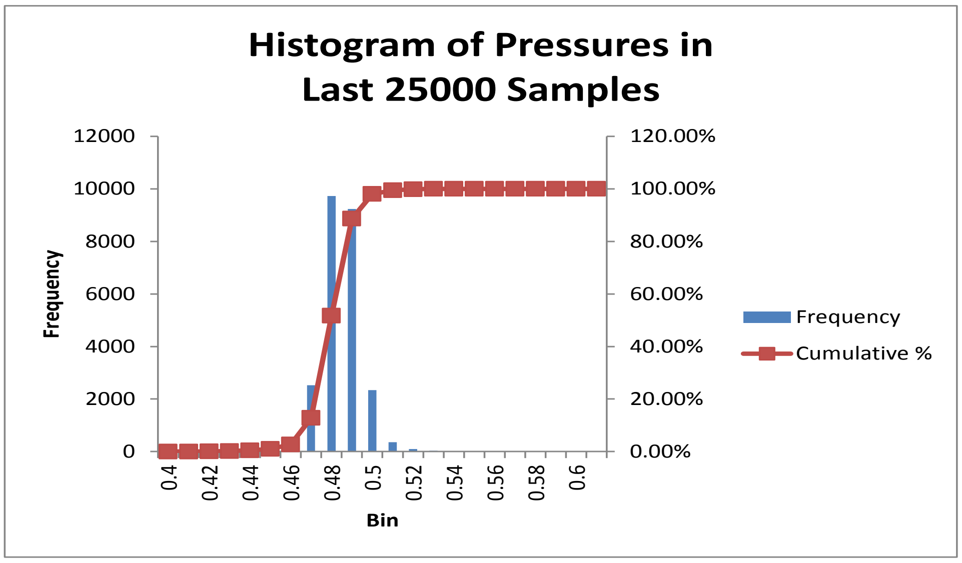
Return now to the class of "inverse problems" mentioned in Figure 1. An important inverse problem is that of code calibration. Given a model (e.g., a "systems" code) that depends on tuning parameters whose values are uncertain, and observed results that bear on the quantification of those parameters, it is possible to update the probability distributions of the parameters using a Bayesian MCMC approach (in fact, one obtains a *joint* posterior density on the variables). For realistic problems, even today, the computational demands of MCMC imply a need for a surrogate such as a GPM. Work of this general kind is ongoing at many institutions.

In addition to applications such as code calibration, it may be of interest to illustrate the potential of this process (inverse problem using a surrogate) for helping to answer questions about what is *currently* going on in a plant of whose state we are largely ignorant, owing to lack of detailed information. Mathematically, such a process looks a great deal like code calibration, except that we would be ill-advised to use prior distributions for the inputs that overstate our knowledge of the current plant state.

The idea here is to analyze uncertain variables during an actual event, given a calibrated model and a surrogate trained on that model. In the case of the PCCS model analyzed earlier, X1 (leak area) at least is clearly aleatory (we would expect leakage to vary from plant to plant, and perhaps with age). X4 could be regarded as epistemic if we believe that it is determined by the piping characteristics and geometry.

Figure 4 shows the results of applying the same surrogate used above to answer the question: What can we say about X1 and X4, given that pressure is .48 ± .003 MPa ? The figures are based on 100000 samples initialized with a guess of X1=.2, X4= 0.1; X2=X3=X5 are locked down at zero for purposes of illustration. In Figure 4, we see a histogram of the last 25000 samples, well centered at the observed pressure, and with a width determined largely by the predictive variance of the surrogate (the observational error having been taken to be smaller, again for purposes of illustration).

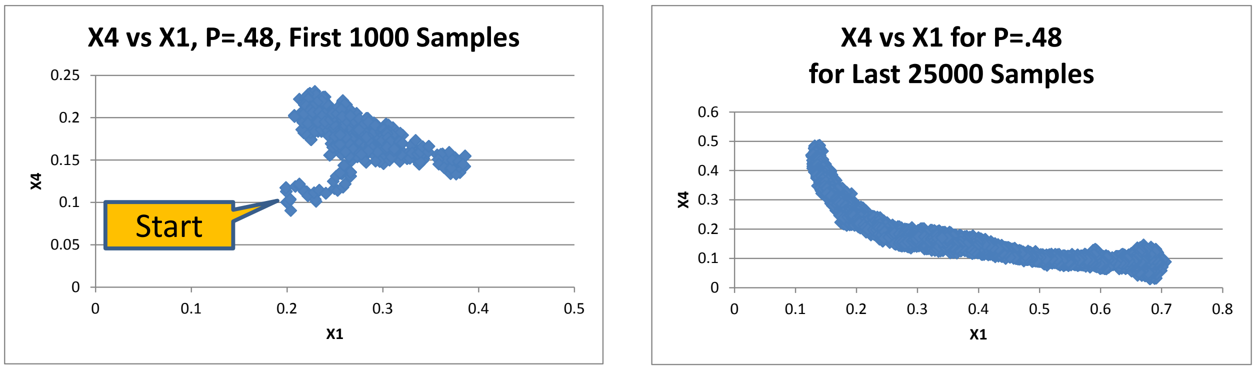
Figure 4. Convergence of the Sampling Process Around P=0.48 MPa.



In Figure 5, we see a comparison of the first 1000 samples with the last 25000 samples. In the plot on the left, from the starting point (a fair but not excellent guess), the process moves up and to the right. In the right-hand portion, we see the samples obtained much later. This is essentially a blurred portion of a slice through the limit surface itself (refer back to Figure 2, with X2=X3=X5=0). The blurring is related to the predictive variance of the surrogate (and, had we chosen to input a larger observational error, that error would influence it as well). The bulge at the right corresponds to the anomaly seen in the high-X1, low-X4 corner of Figure 2. This is close to being a joint distribution of X1 and X4.

It should be clear that had we constrained X4, either based on current observational data or strong prior evidence, we would obtain a significantly localized distribution on X1, the point being that such knowledge could hypothetically inform an actual accident management decision.

Figure 5. Evolution of MCMC Samples



The choice of P=.48 (the limit pressure) was made for this illustration because (in light of the earlier discussion) this is where we know the surrogate's predictive variance to be least poor, owing, we believe, to the training set having been focused deliberately on the limit surface. The results of a similar exercise for pressures not near the limit surface were appreciably less interpretable (they had more scatter, and less visible structure, than Figure 5. For "forensics," and arguably for instances of the inverse problem in general, a much, much better training set than the one used here is needed. "Results" may, of course, emerge regardless of the quality of the training set, but if the predictive variance is large, they will have a great deal of scatter.

6. CONCLUSIONS

This paper compares the performance of different code surrogates in analysis of a set of model results for a passive containment cooling system, originally performed to support quantification of the functional unreliability of that system. The original work made use of an Artificial Neural Network (ANN) as a surrogate for the functional relationship between containment pressure and the issue space variables. Later, the data set was reanalyzed using Alternating Conditional Expectations (ACE). Here, the data set was reanalyzed yet again using a Gaussian Process Model (GPM). All three surrogates yield roughly comparable results for the functional unreliability, based on the input variable pdfs used in the original work. Additionally, the GPM offers information about its predictive uncertainty, based on the training set and, to some extent, how its covariance structure is modeled. This is a well-known and important property of GPMs. Finally, the same surrogate used in the *forward* quantification of functional unreliability was used to illustrate a simple *inverse* problem, taken notionally to be "forensics:" given an observed pressure, what can be said about the inputs? It was found that intelligible results can be obtained for pressures near where most of the training runs occurred, but not necessarily for other regions. This suggests that a much better training set is needed for inverse problems.

Acknowledgements

This work was carried out under DOE Idaho Operations Office Contract DE-AC07-051D14517.

7. REFERENCES

1. For an introduction to GPMs, see (for example) C. E. Rasmussen and C. K. I. Williams, **Gaussian Processes for Machine Learning** (MIT, 2006).
2. J. J. Vandenkieboom, "Reliability Quantification of Advanced Reactor Passive Safety Systems," Ph.D. Thesis (University of Michigan, 1996).
3. Z. Cui, J. C. Lee, J. J. Vandenkieboom, and R. W. Youngblood, "Unreliability Quantification of a Containment Cooling System through ACE and ANN Algorithms," *Trans. Am. Nucl. Soc.* **85, 178 (2001).**
4. J. C. Lee and N. J. McCormick, **Risk and Safety Analysis of Nuclear Systems** (Wiley, 2011). See especially §11.2.4.