SAND2020-1357C

Proceedings of the 28th Annual International Conference on Nuclear Engineering
ICONE28-POWER2020
August 2-6, 2020, Anaheim, USA

ICONE28-POWER2020-16194

UNCERTAINTY QUANTIFICATION FOR CYBER-PHYSICAL PWR EXPERIMENTS

Taylor K. McKenzie
Cyber Resilience R&D
Sandia National Laboratories
Albuquerque, NM 87185
Email: tmcken@sandia.gov

Thomas D. Tarman
Experimental Cyber Initiatives
Sandia National Laboratories
Albuquerque, NM 87185
Email: tdtarma@sandia.gov

Christopher Lamb
Energy Security
Sandia National Laboratories
Albuquerque, NM 87185
Email: cclamb@sandia.gov

ABSTRACT

Uncertainty quantification (UQ) is the process of identifying how uncertainty in inputs to a simulation propagate to uncertainty in outputs from that simulation. When a simulation is computationally intensive, a surrogate function is commonly used to serve as a proxy for that simulation, making it feasible to combine results from the simulation and surrogate function to efficiently sample from the simulation many times. There exist many methods to construct surrogate functions for simulations of physical systems (e.g., computational fluid dynamics), but applications to cyber-based experiments are less understood. We explore the suitability of various surrogate functions against a cyber-physical emulation and simulation of a pressurized water reactor (PWR) that is subjected to various cyber-focused disruptions. Specifically, malicious traffic is injected to heater and sprayer actuators, and it is determined from the model whether pressure exceeds a critical threshold within the period of the disruption. We assume that which actuators are targeted and values injected to targeted actuators are uncertain, random variables following specified distributions. We investigate the performance of traditional surrogate function formulations (e.g., polynomial chaos expansions and Gaussian processes) and find that the assumption of conditional determinism of outputs in these methods is violated in our cyber experiments. We then develop a Gaussian process model that accounts for measurement error and fit it to the binary outcome of whether the critical pressure threshold was crossed using a limited number of simulations. Using this surrogate function, we identify effects of input parameters and determine which variables drive higher frequency of undesirable outcomes. Finally, using specified distributions of input parameters and the surrogate function, we derive an overall probability that the pressure will exceed the critical threshold in a disruption scenario.

Sandia National Laboratories is managed and operated by NTESS under DOE NNSA contract DE-NA0003525.

1 INTRODUCTION

Cyber-experimental systems, such as virtual machine testbeds and discrete event simulations, are used to assess system responses to changing conditions in a carefully controlled manner. In reality, experimentation on live systems is often infeasible because it can disrupt the operations the cyber systems are intended to perform, so experimentation in emulation and/or simulation domains allows an analyst to explore scenarios in a safe, offline environment.

When a cyber experimentation framework is used to evaluate high consequence systems, the analyst needs rigorous experimental approaches that are repeatable and account for uncertainty. Uncertain inputs that represent environmental conditions, configurations, or threats will lead to uncertainty in cyber system responses such as connectivity, available bandwidth, or physical effects (for cyber-physical systems). Assessing how these uncertain model inputs propagate to system responses allows the analyst and decision maker to understand these uncertainties, determine which uncertainties are important, and make better decisions about system design and cyber security mitigations.

This study assesses the cyber-physical response of a hypothetical pressurized water reactor (PWR) system under cyber attack. It accounts for uncertainty in the attack details (i.e., input values for heater and sprayer actuators) and their effects on a response of interest (i.e., differential pressure). Because emulation-based experiments run in real time, replicated experiments are time intensive, so this study also evaluates surrogate functions that can be effectively used to efficiently compute replicate results with statistics that best match those generated by an emulation-based model.

This paper describes the UQ experimental processes for this scenario and results. Section 2 provides background on UQ in general, the use of surrogate functions to represent cyber systems, and emulation-based testbeds for conducting high-fidelity experiments. Section 3 provides additional background on surrogate functions, including the selection of a specific class of surrogate functions (Gaussian Processes) to model cyber systems. Section 4 describes the PWR application, the cyber model, using GP to model the system, and results. Limitations and extensions of the analysis are covered in Section 5. Concluding remarks are offered in Section 6.

2 BACKGROUND

2.1 UNCERTAINTY QUANTIFICATION

Uncertainty is inherent in the modeling process and can arise from an array of sources. There can be uncertainty in inputs to the model, parameters of the model, or even the very form of the model itself. While a chief concern in modeling is to produce unbiased predictions of outcomes, it is often also important to properly quantify uncertainty in predictions from whatever sources it may arise. Broadly, uncertainty quantification (UQ) focuses on identifying key sources of uncertainty and determining how that uncertainty propagates through a system or model to ultimate outcomes of interest.

Berger and Smith describe a framework to consider various sources of uncertainty [1]. They consider a real process that depends on inputs x, where the output of that process is given by $y^R(x)$. High-fidelity computational models are used to simulate or emulate the real process to produce predictions of output, denoted $y^M(x;\varphi,\beta_\varphi)$, where φ is the form of the model and β_φ are parameters of the model. The form and parameters of that computational model are trained using observations of outputs, denoted $y^O(x) = c(y^R(x),\varepsilon)$, where ε is noise and c is a function applying noise to the deterministic output $y^R(x)$. Finally, for computationally-intensive models, a surrogate function, described in the following section, may be used to approximate the computational model. Thus, uncertainty about the true output can arise from the following sources.

1. Uncertainty due to observational noise, ε.
2. Uncertainty about the form φ and parameters β_φ of the computational model and thus whether y^M is unbiased for y^R. This uncertainty arises due to observational noise, among other factors.
3. Uncertainty about whether the surrogate function accurately describes the computational model.
4. Uncertainty about the values of inputs x, which contributes to the distribution of outcomes.

UQ analyses aim to measure the effect of some or all of these sources of uncertainty on output of the real process. Depending on the focus of the study and nature of the process, some potential sources of uncertainty may be excluded. Ultimately, a UQ analysis will propagate relevant uncertainties to produce a description of the distribution of outcomes.

Under relatively restrictive assumptions, it may be possible to arrive at closed-form solutions for outcome distributions; however, doing so is frequently intractable for most realistic scenarios. As a result, approximate methods like Monte Carlo (MC) analysis are frequently employed to propagate uncertainties and form an approximation of output distributions [2]. For each of a specified number of iterations, inputs and parameters are drawn from their respective distributions and fed through the model or system, and outputs are subsequently recorded. By repeating this process many times, it is possible to form an approximate output distribution.

2.2 SURROGATE FUNCTIONS

UQ analyses have traditionally been applied to physics-based or other similarly computationally-intensive simulations. In such scenarios, evaluating the model for a given set of inputs and parameters may take an appreciable amount of time, and it may be infeasible to evaluate the model over enough repetitions to form an adequate description of the output distribution. A frequently-used, computationally-feasible alternative is to create a relatively simple function that can approximate the distribution of outputs given the distribution of inputs [3]. Observed outputs from a limited number of actual simulation evaluations can be used to build these surrogate functions, which can then be called many times to efficiently build a more complete output distribution.

Broadly, surrogate functions can take on one of a few forms to address specific analysis goals, listed below. We denote random inputs to the simulation with X and possibly random outputs from the simulation with Y.

1. A possibly random function f with the property that the distribution of $f(X)$ is approximately the same as the distribution of Y.
2. A deterministic function g with the property that $g(x)$ is approximately the same as $E[Y|x]$ for all x in the range of X.
3. A necessarily random function h with the property that the distribution of $h(x)$ is approximately the same as the distribution of $Y|x$ for each x in the range of X.

Historically, surrogate functions were used with computationally-intensive deterministic simulations. In this application, the surrogate function would be trained on output from a small number of simulations using various input conditions and then used to predict output for other input values. Thus, surrogate functions have traditionally focused on producing unbiased predictions, as in (2) above, with the possibility of also estimating uncertainty in those predictions [1]. Many surrogate functions have been developed to address this type of problem, including traditional Gaussian process models, regression splines, neural networks, and many others. In addition, some methods developed primarily to address (1) above, such as polynomial chaos expansion, can also produce unbiased estimates of the conditional mean of output, as in (2). Surrogate functions of form (2) are also especially useful in explaining how changes in inputs affect mean values of outputs, offering a means to, for example, select input values to optimize outputs.

There has been significant research in developing methods to validate surrogate functions and select which type of function is most appropriate for a given application [4]. These methods tend to use split-sample validation and loss functions that are focused on accuracy of mean predictions, such as mean squared error. As a result, many existing validation and selection methods tend to be best suited towards to surrogate functions of form (2). Other methods make use of statistical tests of distributional similarity, such as the Kolmogorov-Smirnov test, and are well-suited to determining the performance of surrogates of form (1).

Surrogate functions of form (3) have not been developed and utilized to the same degree as those of forms (1) and (2), potentially due to computational complexity and lesser need for such detailed results [5]. However, surrogate functions of form (3) can be very powerful; they are able to accomplish the goals of forms (1) and (2) by construction and can offer insights on how mean, variance, and higher moments of outputs are affected by changes in inputs. Many surrogate functions of form (2) are readily able to be extended to produce distributional predictions rather than point predictions. For example, under the assumption of conditional normality of outputs, predictions from linear regressions are asymptotically normal; also, input uncertainty can be propagated through Gaussian process models account for measurement error, as described in the following section. Further, frameworks to validate and select surrogate functions can be also be applicable to functions of form (3) by choosing an appropriate loss function, such as likelihood-based losses.

This research investigates a Gaussian process model that has been extended to produce predictions of distributions rather than point predictions. While we qualitatively judge the performance of this surrogate, we do not carry out formal validation nor do we investigate other surrogate functions. However, this research can be readily extended to more thoroughly consider surrogate functions of form (3).

2.3 EMULATION AND VIRTUAL TESTBEDS

Cyber experimentation can be performed on live systems, emulated virtual machine testbeds, and/or using discrete event simulators. Live system experimentation is often used when details about the system hardware and software are highly relevant to the results that are produced (i.e., high hardware and software fidelity). However, experimentation on the live system is expensive and doesn't scale well (due to hardware costs), and runs the risk of impacting the functions it is intended to perform.

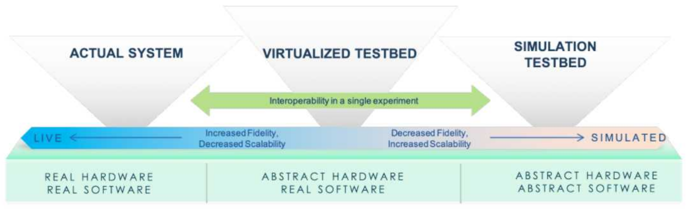
ACTUAL SYSTEM
VIRTUALIZED TESTBED
SIMULATION TESTBED
Interoperability in a single experiment
LIVE
Increased Fidelity, Decreased Scalability
Decreased Fidelity, Increased Scalability
SIMULATED
REAL HARDWARE REAL SOFTWARE
ABSTRACT HARDWARE REAL SOFTWARE
ABSTRACT HARDWARE ABSTRACT SOFTWARE

FIGURE 1: Spectrum of Cyber Experimentation Approaches

Simulation (using a discrete event simulator such as ns3[1]) is a software-based experimentation approach that uses software models to represent hardware and software functions such as network interfaces, applications, and network protocol stacks. They are lightweight software models, so they scale very well and are inexpensive to deploy. However, hardware and software details are abstracted, so depending on the experimental question and amount of effort invested in developing these models, their level of fidelity can vary (but is always lower than live system testing).

Emulation-based testing, using virtual machine technologies such as QEMU and tools to design and orchestrate experiments, provide a compromise between fidelity and scalability. With virtual machine environments, hardware platforms are abstracted into the virtual machine hypervisor, however the operating systems and application software that runs in these virtual machines is the same software that runs on physical machines. Therefore, one can expect high fidelity in software, and reduced fidelity in hardware.

Cyber experiments can also use a combination of emulation, simulation, and "hardware in the loop" if an experiment requires a degree of scalability as well as hardware realism. In the scenario described here, a combination of emulation and simulation is used - emulation to model cyber components (hosts, switches, routers, and industrial control devices), and simulation is used to model the PWR processes. Emulation is an appropriate selection for the cyber components because the experimental questions are relevant to software processes (networking, control processes, applications, and attacker actions). Simulation is appropriate for the PWR processes because they can be expressed mathematically and have been independently validated as representative of the actual system.

[1]https://www.nsnam.org

3 SURROGATE FUNCTIONS, GAUSSIAN PROCESSES, AND RANDOM OUTCOMES

As mentioned previously, surrogate functions were initially developed as a proxy for computationally-intensive deterministic simulations. In that paradigm, researchers evaluate the simulation a limited number of times using various input conditions, then use recorded output from those evaluations to fit a surrogate function that can be used to predict or otherwise generate output for different input values. In this section, we will explore the Gaussian process (GP) surrogate function using example data, first under the assumption of deterministic outputs then relaxing that assumption to allow for observational noise. While this research only considers GPs, similar formulations could be developed for other surrogate functions.

For the remainder of this section, we will consider generating representative output from the function $w(x) = \sin(x) * x$. The standard GP model, which we use to predict w, has the following form.

$$f(x) \sim N(0, K_\theta(x)) \tag{1a}$$

$$y \sim N(f(x), \sigma^2 I_N). \tag{1b}$$

In these equations, θ and σ are parameters of the model and N represents the number of output observations used to train the model. Equation (1a) represents the prior over values of $f(x)$, formed using kernel function K_θ. A common choice of kernel function is the normal kernel, also referred to as the exponentiated quadratic kernel. For univariate inputs,[2] as in our example, this kernel function takes the form

$$K_\theta(x)_{ij} = \alpha^2 \exp\left(-\frac{1}{2p^2}(x_i - x_j)^2\right) \quad \forall i = 1,...,N; j = 1,...,N, \tag{2}$$

where α and p are a positive scalars (contained in the parameter vector θ) and $x_i \in \mathbb{R}$ is the ith observation of x, of which there are N total observations. This implies that $K_\theta(x)$ is an $N \times N$ positive definite matrix. Finally, eq. (1b) relates observations of output, y, to their predicted values, $f(x)$.

Traditionally, the parameters θ and σ have been fit using maximum likelihood estimation (MLE). Parameter point estimates representing a single realization of the GP are then used for inference and prediction. Specifically, if one wants to use N^* new input values, contained in the vector x^*, to predict output y^*, the following result can be used [6].

$$y^* | x^*, y, x, \theta, \sigma \sim N(A, B), \tag{3}$$

[2] A similar formulation can be derived for multivariate inputs using the multivariate version of the normal kernel.

where

$$A = K_\theta(x^*, x)\Sigma^{-1} y \tag{4a}$$

$$B = K_\theta(x^*) - K_\theta(x^*, x)\Sigma^{-1} K_\theta(x^*, x)', \tag{4b}$$

where

$$\Sigma = K_\theta(x) + \sigma^2 I_N. \tag{5}$$

Using a normal kernel, the kernel functions above are defined as

$$K_\theta(x)_{ij} = \alpha^2 \exp\left(-\frac{1}{2p^2}(x_i - x_j)^2\right) \quad \forall i = 1,...,N; j = 1,...,N, \tag{6a}$$

$$K_\theta(x^*)_{ij} = \alpha^2 \exp\left(-\frac{1}{2p^2}(x_i^* - x_j^*)^2\right) \quad \forall i = 1,...,N^*; j = 1,...,N^*, \tag{6b}$$

$$K_\theta(x^*, x)_{ij} = \alpha^2 \exp\left(-\frac{1}{2p^2}(x_i^* - x_j)^2\right) \quad \forall i = 1,...,N^*; j = 1,...,N. \tag{6c}$$

In the maximum likelihood estimation of the above GP model, all data points used to train the model will necessarily be fit exactly so long as each observation of inputs is unique [5]. In other words, since predictions are formed using a single realization of the GP, this model formulation does not accommodate any kind of measurement error.

As an illustration, five values of x were uniformly sampled from the interval $[0, 10]$, and values of $y = w(x) = x\sin(x)$ were recorded and used to fit the GP model. Actual values of the function, points used to fit the GP, mean predictions, and the 95% confidence interval of predictions are shown in fig. 2. As can be seen, the model infers zero uncertainty at observed data points, low uncertainty near observed data, and greatest uncertainty far from observed data.

This estimation and prediction scheme fails to propagate uncertainty in parameter estimates, which could be caused by observational noise, through to uncertainty in predictions. As a result, this framework is inappropriate when observational noise may exist. To illustrate, 100 values of x (denoted x_i for $i = 1, ..., 100$) were uniformly sampled from the interval $[0, 10]$, and values of $y_i = w(x_i) = \sin(x_i) * x_i + \varepsilon_i$ were simulated and recorded for each i, where $\varepsilon_i \sim$ iid $N(0, 1)$. Ten of those data points were randomly selected to train the GP model; the remainder were used to illustrate validation of the model. A plot of the

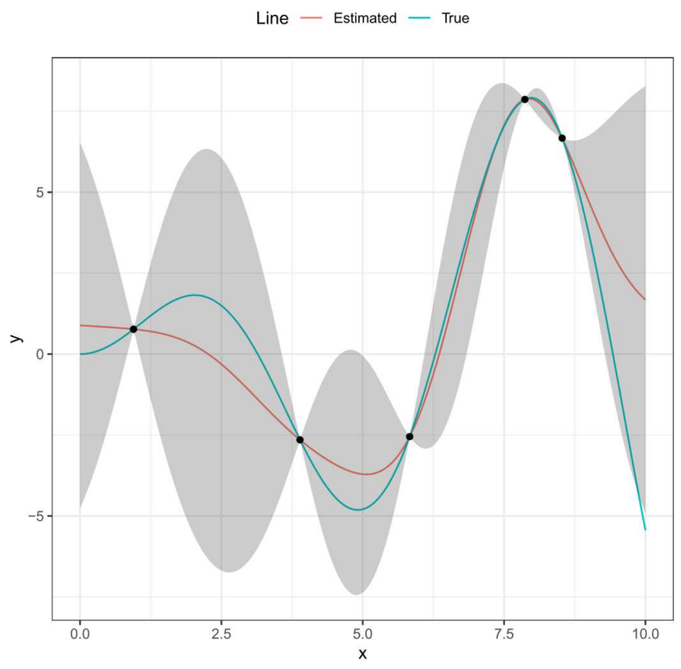

FIGURE 2: Traditional GP Fit to Deterministic Outcomes

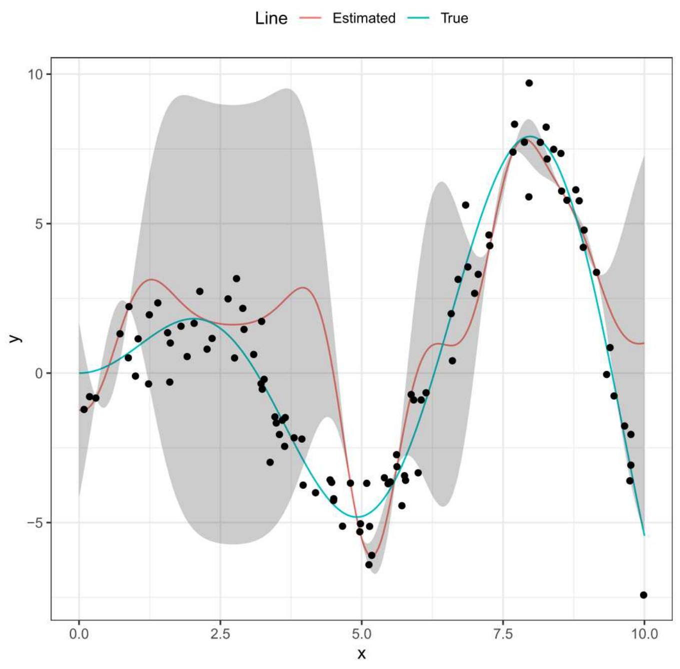

FIGURE 3: Traditional GP Fit to Random Outcomes

expected value of *w*, all simulated data points, mean predictions, and the 95% confidence interval of predictions is shown in fig. 3. From this plot, it is clear that the traditional GP predictions have a tendency to overfit the data and can severely underestimate uncertainty, especially near observed data points.

It can be very difficult to analytically propagate uncertainty through a GP model in a classical estimation framework due to non-linearities in the model. Bayesian estimation of a GP, on the other hand, utilizes Monte Carlo methods that make propagation of parameter uncertainty more feasible at the cost of being more computationally expensive. In addition to the model specification above, a Bayesian framework additionally requires specification of prior distributions over the parameters. When there is little knowledge about likely parameter values, it is best practice to specify priors that are diffuse to avoid influencing ultimate results. Gelman advocates using a half-Cauchy prior for parameters related to variance, such as α, p, and σ in our model [7]. For this illustration, we use the following priors:

$$\alpha \sim \text{Cauchy}^+(0,10) \tag{7a}$$
$$p^2 \sim \text{Cauchy}^+(0,10) \tag{7b}$$
$$\sigma \sim \text{Cauchy}^+(0,10). \tag{7c}$$

We fit this model by sampling from parameters' posterior distributions utilizing the Stan platform [8]. Predictions are then formed using eq. (3) through eq. (6c) for each parameter sample. When predictions using each parameter sample are combined, an approximate posterior distribution of predictions is formed that includes both uncertainty due to missing information about *w* and uncertainty about parameter values caused by observational noise. As a final illustration, the GP model was fit again using the same training and validation data used in fig. 3, this time under the Bayesian framework. A plot of the expected value of *w*, all simulated data points, mean predictions, and the 95% confidence interval of predictions is shown in fig. 4. It is clear that the Bayesian formulation more properly quantifies uncertainty of predictions and doesn't suffer from the same degree of overfitting compared to the traditional GP formulation when observational noise is present.

4 NUCLEAR POWER PLANT APPLICATION

The system described here is a hypothetical cyber-controlled pressurized water reactor (PWR) system. [3] The scenario assumes that an attacker has a presence on a network host, and has the ability to connect to an actuator and command it with actuation values to affect the differential pressure in the reactor system. The experimental question that this model is intended to answer is: given uncertainty in the actuator values that the

[3]The models and scenarios are not intended to describe real facilities, attacks, or malware. They represent hypothetical scenarios and are included only for illustrative purposes.

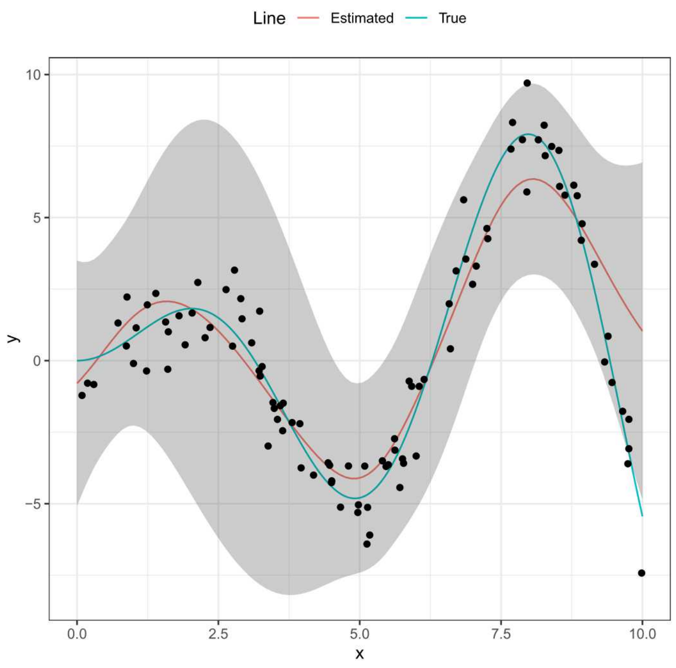

FIGURE 4: Bayesian GP Fit to Random Outcomes

attacker selects, what is the corresponding uncertainty in 1) the absolute pressure values, and 2) the time it takes the pressure to exceed a threshold.

4.1 SYSTEM, IMPLEMENTATION, AND DISRUPTION

As described earlier, the model consists of an emulation-based model of the system's cyber components, and a numerical model of the PWR process. The numerical model (implemented in Simulink) represents a pressurizer, which regulates the pressure in the primary reactor coolant loop of a PWR and consists of two regions - an upper steam region (a steam "cushion"), and a lower water region. The upper steam region has a continuous steam region with discrete water droplets and wall condensate that fall into the lower water region. Conversely, the lower water region has a continuous water region with a discrete vapor region that rises into the vapor region [9] [10] [11] [12]. Attached to the hot leg of a PWR, the pressurizer is comprised of electric variable heaters, electric backup heaters, and a spray valve to dynamically control system pressure. If the pressurizer pressure control system actuates either the electric heaters or spray valve the water level in the pressurizer will change. To maintain the desired water level in the pressurizer the pressurizer level control system utilizes charging pumps and letdown valves to vary the water inventory in the reactor coolant system [13].

The emulation-based model for the cyber components, shown in fig. 5, is comprised of a number of virtual machines that implement a supervisory control and data acquisition (SCADA) server, a router, network switches, and remote terminal units (RTUs) and programmable logic controllers (PLCs) that sense pressure and actuate a heater and sprayer. These virtual machines are configured into a network topology and orchestrated using minimega. [4]

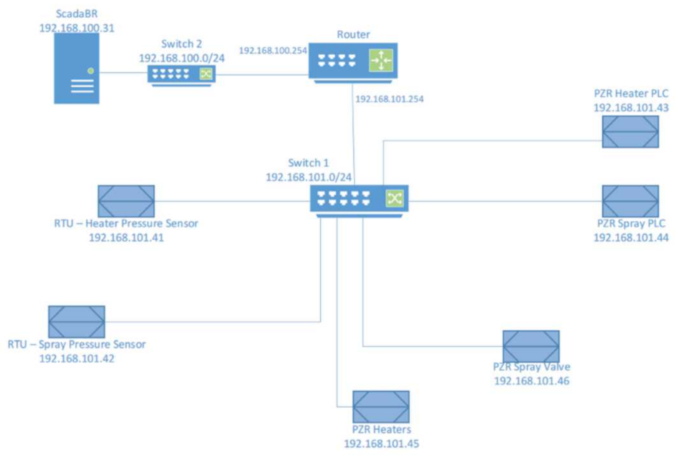

FIGURE 5: Emulation Model Topology

An attacker process runs on the minimega host machine and injects false heater and sprayer values. This process connects to the heater and sprayer actuators and creates malicious MODBUS traffic to send these values to the targeted device(s), as specified in command line parameters. Exposing these values to the command line allows an external tool such as DAKOTA [14] to select these values as part of a systematic series of experiments, such as uncertainty quantification, optimization, or mapping the system responses.

Data is collected throughout the experiment. When the experiment is started, baseline data is collected during the first five minutes, before the attack is started, to allow the system to stabilize. Once the attack is started, it is allowed to proceed for two minutes. Data is collected during the attack phase as well, and once the experiment is concluded, analyzed to determine if and when the pressurizer value crosses the threshold.

4.2 QUANTITIES OF INTEREST AND EMPIRICAL MODEL

Severe consequences can occur if the differential pressure of the heater or sprayer exceeds one MPa. This UQ analysis focused on the probability that differential pressure would exceed one MPa at any point over a two minute attack conditional on which actuators are targeted and what values are injected to targeted actuators as a part of the attack. These probabilities could

[4]https://minimega.org

be estimated directly by repeatedly running the emulation over the range of input values. However, as previously mentioned, the emulation is computationally- and time-intensive, making it difficult to obtain enough samples to accurately predict outcomes with adequate fidelity. Surrogate functions provide a method to efficiently approximate outcomes; however, since the outcomes of emulation runs are necessarily random, methods must accommodate observational noise. We use a variant of the GP model developed in section 3, adapted to describe binary outcomes (e.g., whether or not differential pressure exceeds one MPa).

Let X represent a continuously-valued matrix of inputs with N rows/observations and k columns/variables. Suppose y is a binary outcome, where $y = 1$ represents "success" and $y = 0$ represents "failure." Then, GP model takes the following form:

$$\alpha \sim \text{Cauchy}^+(0,10) \tag{8a}$$
$$P_{ii} \sim \text{Cauchy}^+(0,10) \qquad \forall i = 1,...,k \tag{8b}$$
$$P_{ij} = 0 \qquad \forall i \neq j \tag{8c}$$
$$f(X) \sim N(0, K_{\alpha,P}(X)) \tag{8d}$$
$$y \sim \text{Bernoulli}\left(\sigma^{-1}(f(X))\right), \tag{8e}$$

where P is a $k \times k$ positive definite matrix, σ^{-1} is the inverse logit function, defined by $\sigma^{-1}(x) = 1/(1+\exp(-x))$, and $K_{\alpha,P}(X)$ is the multivariate normal kernel, defined by

$$K_{\alpha,P}(X)_{ij} = \alpha^2 \exp\left(-\frac{1}{2}(X_i - X_j)' P^{-1} (X_i - X_j)\right) \qquad \forall i = 1,...,N; j = 1,...,N, \tag{9}$$

where X_i is the ith observation/row of X.

Since which actuators are targeted is not a continuously-valued variable, it cannot be included into inputs in X. Instead, we fit a separate GP model for each of three cases: (1) where the heater actuator is targeted, (2) where the sprayer actuator is targeted, and (3) where both heater and sprayer actuators are targeted. The input matrix X contains values injected to targeted actuators. Input data were generated using Latin hypercube sampling as implemented in Dakota, fed to the emulation, and the time elapsed before the heater or sprayer differential pressure exceeded one MPa in magnitude (if that event occurred) was returned to Dakota and recorded. A total of 250 samples from the emulation were used to fit the GP surrogate function; 83 samples targeted the heater actuator, 83 targeted the sprayer actuator, and 84 targeted both actuators. The binary outcome y was constructed to take a value of one if the differential pressure of either heater or sprayer actuators exceeded one MPa within the two minute attack and a value of zero otherwise.

The GP model was fit using the Stan platform [8] using 1000 warmup iterations and 1000 sampling iterations. Trace plots were examined to ensure approximate convergence of the Markov chain to the posterior distribution. Further, initial conditions of the Markov chain were varied to ensure non-reducibility assumptions were justified. Simulations of predicted values of $f(X^*)$ were generated using eq. (3) for specified input values X^*; probability that differential pressure would exceed one MPa was then simulated by passing simulated predictions through the inverse logit function. Output of the GP is therefore the distribution of probabilities that the one MPa threshold would be crossed conditional on which actuators are targeted and what values are injected to targeted actuators.

As an exception, there were no simulations targeting only the heater that resulted in differential pressure exceeding one MPa. Thus, the posterior of the GP model was very unstable to the point where the Markov chain had extreme difficulty in convergence. Instead, an alternative formulation that did not depend on values injected to the heater was used:

$$p \sim \text{Beta}(1,1) \tag{10a}$$
$$y \sim \text{Bernoulli}(p), \tag{10b}$$

where $\text{Beta}(1,1)$ is the Beta distribution with shape and rate equal to one (i.e., the uniform distribution on $[0,1]$). The posterior distribution is then given by

$$p \sim \text{Beta}(s+1, f+1), \tag{11}$$

where s is the number of successes in the data (i.e., where $y = 1$) and f is the number of failures (i.e., where $y = 0$). In our emulation data where the heater actuator was targeted, $s = 0$ and $f = 83$, so the posterior is given by $p \sim \text{Beta}(1,84)$. This formulation allows for the possibility that $p > 0$ because we simply have not observed enough observations for an instance where $y = 1$ to occur.

4.3 RESULTS

As mentioned previously, three separate GP surrogate models were fit to emulation data: one where only the heater actuator was targeted, one where only the sprayer actuator was targeted, and one where both heater and sprayer actuators were targeted. Plots of mean predicted probability of differential pressure exceeding one MPa in magnitude and 95% interior intervals for each case are shown in fig. 6, fig. 7, and fig. 8. A few general conclusions can be drawn from these results. First, the expected probability of reaching the critical threshold is relatively low for injected sprayer values below about 40 and quickly becomes very likely above that level, especially when only the sprayer is targeted.

Second, the probability of differential pressure crossing the critical threshold is generally higher for injected sprayer values less than 40 when both heater and sprayer actuators are targeted compared to when only the sprayer actuator is targeted.

Finally, predictions from the GP models show some evidence of overfitting, driven in large part by noisiness in the data. For example, there are several dips in predicted probability of crossing the critical threshold for sprayer values above about 40 when only the sprayer actuator and when both actuators were targeted; these dips coincided with samples where differential pressure did not exceed one MPa, and the GP model inferred that probability of crossing that threshold must be lower around those points. There are several methods that could reduce this overfitting and improve the overall fit of the GP, and some are detailed in the following subsection.

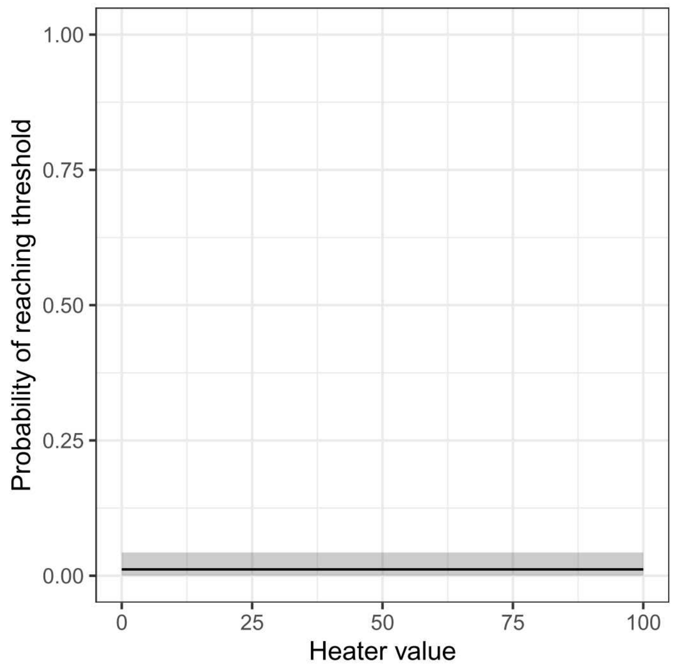

FIGURE 6: Targeting Heater Only

As mentioned previously, in addition to providing insight in how inputs influence outputs, this GP surrogate function can also be used to propagate input uncertainty through to output uncertainty. We assume a design basis threat that targets the heater actuator with probability 1/3, the sprayer actuator with probability 1/3, and both actuators with probability 1/3. We also assume values injected to targeted actuators are uniformly distributed between 0 and 100. Under these assumptions, we estimated the unconditional probability of the critical threshold being crossed. First, the probability of differential pressure exceeding one MPa conditional on the actuator targeted was estimated by drawing 1000 values to inject to targeted actuators from their respec-

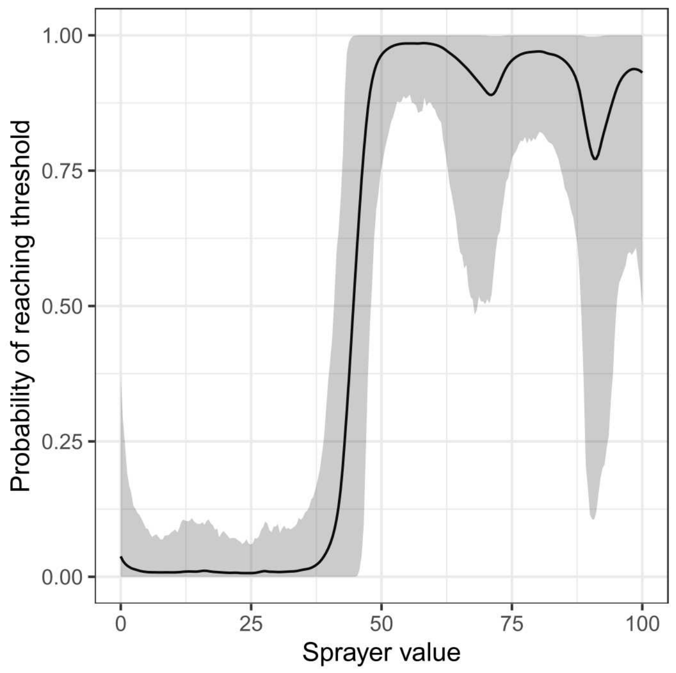

FIGURE 7: Targeting Sprayer Only

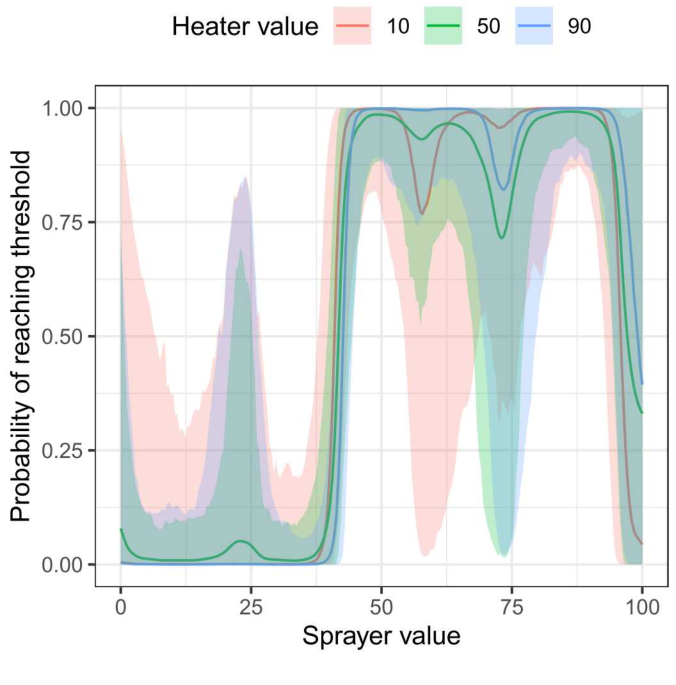

FIGURE 8: Targeting Both Heater and Sprayer

tive distributions, simulating probability of crossing the threshold from the above GP models, and finding the mean of those simulated values. Resulting estimated probabilities are shown between the second and third layers of fig. 9. Unconditional probabilities were then formed by taking the average of conditional probabilities weighted by the probability each actuator would be targeted. Overall, we found the unconditional probability differential pressure would exceed one MPa was approximately 35.7%.

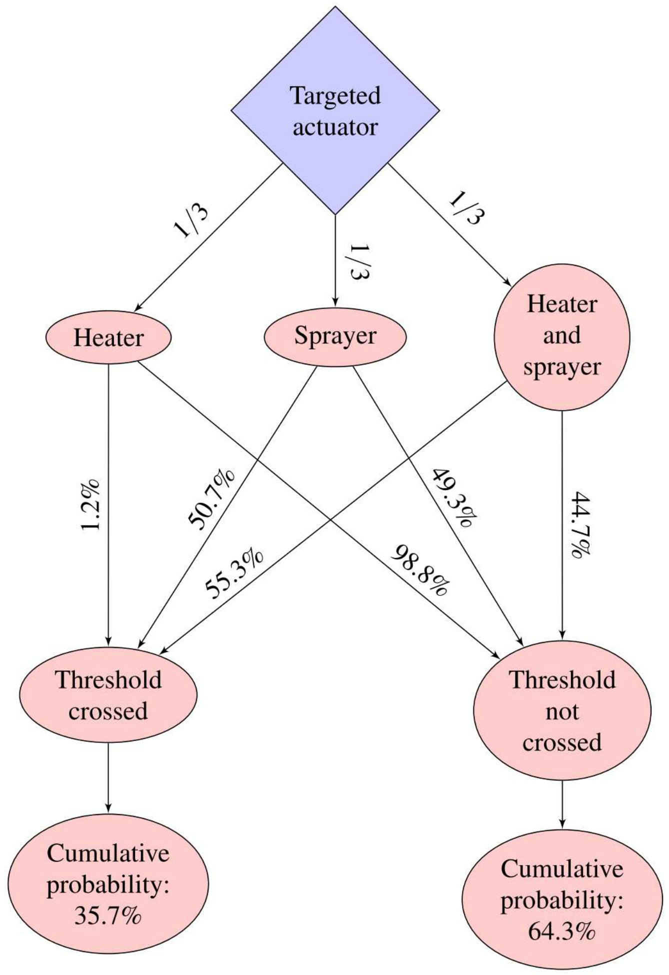

FIGURE 9: UQ Probability Tree

Using a surrogate function provides dramatically lower computational cost per sample compared to running the emulation. Each iteration of the emulation required five minutes to stabilize, two minutes to run the attack, and some additional time to create, start, and clean up the experiment, for a total of at least seven minutes per iteration. In contrast, on a desktop PC with a 3.5GHz processor, each evaluation of the GP takes less than 0.7 seconds on average. Given the noisiness of emulation outcomes, many iterations are required to achieve acceptably precise estimates; the use of the GP surrogate function for this application makes it feasible to obtain these estimates and characterize the noisiness of outcomes.

4.4 EXTENSIONS

The research presented in this paper presents a potential starting point to performing robust UQ on cyber-dependent ICS emulation experiments. However, there are several avenues of research to be explored that can further strengthen this analysis. First, as noted in the previous section, the GP model used in this paper is relatively simple and could certainly be tuned and improved upon to produce better predictions and characterizations of uncertainty. There are several theoretical concepts that could be included into the model. For example, one may expect that the probability of crossing the critical threshold is monotonically increasing in values injected to the actuators, a constraint that can be imposed on the GP. Additionally, since values injected to the heater do not appear to have an effect on the probability of crossing the critical threshold, one may expect the predictions to look similar when the sprayer actuator is targeted compared to when both heater and sprayer actuators are targeted; correlations in those outcomes could be incorporated into the GP model to allow for such a relationship.

The sampling method used to select inputs for the training data was relatively simple and predominantly used to evenly cover the input space. Instead, a more targeted approach could be used to reduce uncertainty in GP predictions and more accurately separate uncertainty due to noise in outcomes and uncertainty about function values due to insufficient sampling.

This research examined a binary outcome, which was relatively straightforward to describe in a GP model. Other outcomes may be real-valued or an otherwise more diverse set of values. In such cases, it is crucial to accurately describe the distribution of outcomes conditional on input values. While standard models typically assume conditional normality of outcomes, our initial investigations showed outcomes may come from more exotic distributions, such as Student's t, Cauchy, or other stable distributions. It is clear that correctly specifying the outcome distribution is critical to correctly predict the conditional distribution of outcomes, but since these distributions also may not satisfy assumptions of the Central Limit and Gauss-Markov theorems, standard methods to predict mean outcomes may not be reliable either.

Finally, this research focused solely on an extended form of the GP model as a surrogate function; there exist many other potential candidates that could provide a better description of outcomes. Future research would include development of these methods to predict conditional distributions, methods to evaluate and compare different surrogate functions of this form, and general rules or guidelines to select which types of surrogate functions to consider for a given system and outcome.

5 CONCLUSION

The use of cyber-experimental systems allows for the evaluation of vulnerabilities and stability of a real system in a safe environment. There are frequently many uncertain conditions in cyber experiments that can affect the ability of the system to respond to a disruption in complex, nonlinear ways. A clear understanding and quantification of how these uncertainties propagate through the system is critical to identifying effective mitigations and assessing risk.

However, cyber experiments are often computationally-intensive, thereby complicating the task of propagating uncertainty. The surrogate function approach offers an approximate solution, where the experiment is evaluated a limited number of times and a function is trained to predict outcomes. We employ this approach to examine the vulnerability of a hypothetical pressurized water reactor (PWR) system to a hypothetical cyber attack. We include both uncertainty about conditions of the cyber system and details of the attack in our analysis. Due to the highly volatile nature of outcomes from this experiment, we develop and utilize an extension of the Gaussian process surrogate function adapted to accommodate noisy outcomes.

We find that the extended Gaussian process surrogate function is able to characterize uncertainty of outcomes conditional on simulation inputs as well as identify relationships between attack characteristics and outcomes of interest, which could potentially shed light on effective mitigations. We use these methods to analyze both the conditional and unconditional uncertainty of critical outcomes that provide insights on whether the hypothetical cyber attack would be successful. Finally, we identify some limitations of the Gaussian process surrogate and this analysis as a whole and consider extensions to improve predictive capabilities and robustness of results.

6 ACKNOWLEDGEMENTS

The authors wish to thank Eric Vugrin and Robert Bruneau for their leadership and thoughtful suggestions, Meg Galiardi, Amanda Gonzales, and Jamie Thorpe for their help with our UQ experiments, Ray Fasano for the PWR Simulink model and model description, and Tim Ortiz and Ryan Kao for the minimega model of the cyber system.

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.

REFERENCES

[1] Berger, J. O., and Smith, L. A., 2019. "On the statistical formalism of uncertainty quantification". *Annual review of statistics and its application,* ***6***, pp. 433–460.

[2] OHagan, A., 2013. "Polynomial chaos: A tutorial and critique from a statisticians perspective". *SIAM/ASA J. Uncertainty Quantification,* ***20***, pp. 1–20.

[3] Najm, H. N., 2009. "Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics". *Annual review of fluid mechanics,* ***41***, pp. 35–52.

[4] Giunta, A., McFarland, J., Swiler, L., and Eldred, M., 2006. "The promise and peril of uncertainty quantification using response surface approximations". *Structures and Infrastructure Engineering,* ***2***(3-4), pp. 175–189.

[5] Hombal, V., and Mahadevan, S., 2011. "Bias minimization in gaussian process surrogate modeling for uncertainty quantification". *International Journal for Uncertainty Quantification,* ***1***(4).

[6] Rasmussen, C. E., 2003. "Gaussian processes in machine learning". In Summer School on Machine Learning, Springer, pp. 63–71.

[7] Gelman, A., et al., 2006. "Prior distributions for variance parameters in hierarchical models (comment on article by browne and draper)". *Bayesian analysis,* ***1***(3), pp. 515–534.

[8] Stan Development Team, 2018. RStan: the R interface to Stan. R package version 2.18.2.

[9] Baron, R., 1973. "Digital model simulation of a nuclear pressurizer". *Nuclear Science and Engineering,* ***52***(3), pp. 283–291.

[10] Abdallah, A., Mariy, A., Rabie, M., and Nagy, M., 1981. "Pressurizer transient dynamic model". *Nuclear Engineering and Design,* ***73***, pp. 447–453.

[11] Saedi, H., and Griffith, P., 1983. "The pressure response of a pwr pressurizer during an insurge transient". *Transactions of the American Nuclear Society,* ***44***, pp. 606–607.

[12] Baek, M., No, C., and Park, I., 2017. "A nonequilibrium three-region model for transient analysis of pressurized water reactor pressurizer". *Nuclear Technology,* ***74***(3).

[13] Wang, P., He, J., Wei, X., and Zhao, F., 2019. "Mathematical modeling of a pressurizer in a pressurized water reactor for control design". *Applied Mathematical Modeling,* ***65***, pp. 187–206.

[14] Swiler, L., Adams, B., and Eldred, M., 2015. *Dakota: Bridging Advanced Scalable Uncertainty Quantification Algorithms with Production Deployment.*