Regular Paper

A Structured Approach for Rapidly Mapping Multilevel System Measures via Simulation Metamodeling

Scott L. Rosen, Christopher P. Saunders, and Samar K. Guharay[*]

The MITRE *Corporation*, McLean, VA 22102

Received 4 March 2013; Revised 13 January 2014; Accepted 28 September 2014, after one or more revisions
Published online 18 December 2014 in Wiley Online Library (wileyonlinelibrary.com).
DOI 10.1002/sys.21290

ABSTRACT

With increasing complexity of real-world systems, especially for continuously evolving scenarios, systems analysts encounter a major challenge with the modeling techniques that capture detailed system characteristics defining input–output relationships. The models become very complex and require long time of execution. In this situation, techniques to construct approximations of the simulation model by metamodeling alleviate long run times and the need for large computational resources; it also provides a means to aggregate a simulation's multiple outputs of interest and derives a single decision-making metric. The method described here leverages simulation metamodeling to map the three basic SE metrics, namely, measures of performance to measures of effectiveness to a single figure of merit. This enables using metamodels to map multilevel system measures supports rapid decision making. The results from a case study demonstrate the merit of the method. Several metamodeling techniques are compared and bootstrap error analysis and predicted residual sums of squares statistic are discussed to evaluate the standard error and error due to bias. © 2014 Wiley Periodicals, Inc. Syst Eng 18: 87–101, 2015

Key words: quantitative systems engineering; simulation metamodeling; preference modeling

1. INTRODUCTION

A paradigm shift in systems engineering (SE) is required to perform real-time and agile quantitative analysis of complex systems with only modest computational resources. The challenge of this problem involves rapidly performing analysis of alternatives, sensitivity studies, identification of critical parameters, and above all, optimization.

*Author to whom all correspondence should be addressed (e-mail: sguharay@mitre.org).

It is well recognized that modeling, in general, is a more affordable, efficient and effective means for systems analysis than test and evaluation, especially in the context of complex, large system analysis. The full merit of simulation modeling is realized for analysis within the SE lifecycle of large, complex systems; the widely used V (vee)-model illustrates the SE lifecycle [INCOSE, 2007]. Simulation modeling enables achieving an insight into the functionality and structural effects of complex systems that possess diverse components and associated subcomponents, for example, technology, social, and political components. Simulation modeling provides the input–output relationships that define a system's architecture and its performance metrics as a function of the elemental characteristics of the subcomponents.

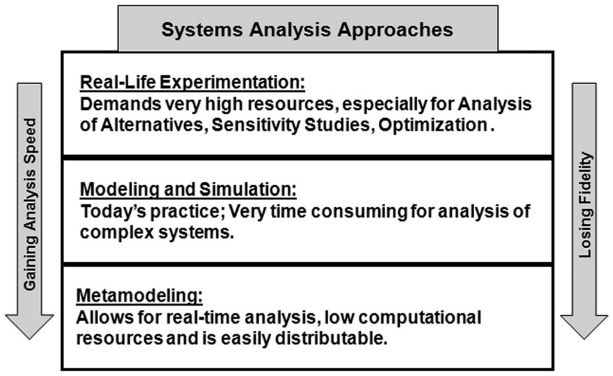
Systems Analysis Approaches
Real-Life Experimentation:
Demands very high resources, especially for Analysis of Alternatives, Sensitivity Studies, Optimization.
Modeling and Simulation:
Today's practice; Very time consuming for analysis of complex systems.
Metamodeling:
Allows for real-time analysis, low computational resources and is easily distributable.
Gaining Analysis Speed
Losing Fidelity

Figure 1. Hierarchy of systems analysis: this represents the key attributes of different approaches. The question of losing fidelity and accuracy arises as the analysis speed increases.

An array of literature exists in the area of model-based systems engineering, also called model based system design [Wymore, 1993; Klir, 1996; Ottino, 2003]. Large scale simulations often require long run times, which can be a hindrance to perform thorough analysis, especially in the agent-based and geospatial domain [Chwif and Paul, 2000]. Executing the simulation on multiple processors in parallel somewhat alleviates this problem, but licensing restrictions may obstruct distribution possibilities on parallel processors. Also, extensive scenario configuration inside the simulation model can add to the long run times making analysis even more challenging. In order to increase its acceptance and utility as a quantitative analytical tool to a general analyst, faster simulation-based analysis is needed. A general systems analyst refers to any type of decision maker, ranging from an acquisition officer to program manager who can interpret systems performance in a quantitative manner.

Different approaches for systems analysis and their key attributes are highlighted in Figure 1. Real-life experimentation has the intrinsic merit to accurately represent system characteristics and avoid assumptions that are often questionable in modeling and simulation. However, this direct approach for systems analysis is the most time consuming and often requires prohibitively high cost in most situations.

Metamodels are used as surrogate models whose primary use is to enable real-time analysis; they are also effective in replacing slow, bottleneck simulations in a federation environment and can be applied to facilitate validation of the large scale simulation. The increase in speed for metamodels comes with a tradeoff to a loss of fidelity from the simulation model. However, the speed at which evaluation can be accomplished with the metamodel enables detailed trade-space analysis. This rapid quantitative analysis, covering the requirements to implementation phases in the V-model, augments the overall systems analysis capability and supports acquisition in the systems development cycle where cost/ benefit relationships must be efficiently and thoroughly determined. While utilizing metamodels as a surrogate model for faster analysis the critical assumption is that the analyst starts with a validated simulation model. From the standpoint of accuracy and reliability the metamodel is, in the best case, as good as the large-scale simulation model that it sampled for calibration.

A recent and comprehensive literature survey on simulation metamodeling is discussed in a review [Barton and Meckesheimer, 2006]. The approaches that have been applied toward simulation metamodeling are: response surface techniques [Myers, 1976; Box, and Draper, 1987; Myers, Montgomery, and Anderson-Cook, 2009], splines [Eubank, 1988; Deboor, 1978; Myers et al., 1996], radial basis functions [Shin, Sargent, and Goel, 2002; Dyn, Levin, and Rippa, 1986; Meghabghab, 2001; Hussain, Barton, and Joshi, 2002], Kriging [Sacks et al., 1989; Kleijnen and Sargent, 2000; Ankenman, Nelson, and Staum, 2010; Staum, 2009; Kleijnen, 2009], Neural Networks [Al-Hindi, 2004; Fonseca, 2003; Lippman, 1987], inductive learning [Michalski, 1983], and genetic programming [Koza, 1992]. Metamodels, in the context of structural reliability encompassing a wide area in engineering, is also discussed in a recent review [Sudret, 2012]. Related to this area of research are developments in systems identification [Le, Bloch, and Lauer, 2001; Ljung, 1999] and phantom system models [Haimes, 2012; Hall, 1989; Horowitz and Lamber, 2006] that are driven by measurements or modeling of experiments. Rosen, Saunders, and Guharay [2012] have recently reported the application of simulation metamodeling to systems consisting of time series inputs and outputs. Osorio and Bidkhori [2012] and Osorio and Chong [2012] have recently shown the benefits of metamodeling for efficient optimization of transportation systems.

A comprehensive definition of a metamodel technique should entail several characteristic factors including the form of the underlying basis functions, the structure for which the basis functions are integrated and the fitting strategy for deriving the metamodel. However, metamodeling techniques are often defined in the literature by only one of these attributes, which can cause confusion. For example, radial basis functions are a type of basis function that is used within a metamodel technique. However, they do not either fully define the metamodel technique itself or fully classify it into a unique family; the basis functions can be expanded or mapped together in many different ways, and even via a Neural Network.

Contained in the method reported in this article is the development of a generalized, simulation metamodeling approach usable by the general systems analyst. The generalized approach provides guidance to the general systems analyst for fitting and validating different metamodeling techniques based on the characteristics of the underlying simulation.

The overall purpose of this metamodeling-based technique is to provide a mapping of the basic SE metrics, namely, measures of performance (MoP) to measures of effectiveness (MoE) to a single figure of merit (FoM) [Hall and McMullen, 2004].

The three SE metrics are defined below:

MoP: Metrics at the component level. *Example*: The detection probability (P_d) of an individual sensor for a scenario, such as security screening.

MoE: Metrics at the system level. *Example*: A system-wide P_d.

FoM: A single measure of overall system effectiveness. Because simulations often consist of multiple system performance measures or MoEs, which can have conflicting preferences or varying significance levels, the FoM can be very effectively and efficiently exploited for decision making.

Example: An aggregation of MoEs with the FoM value as a real number ranging between 0 and 1

The MoP–MoE–FoM mapping is a functional mapping of *m MoP* parameters to a single number representing the overall system effectiveness or *FoM*. It first involves a functional mapping of the *MoP*s to *n MoE*s as in Eq. (1) followed by a mapping of all MoE_i output to a single valued *FoM* as depicted in Eq. (2).

$$MoE_i = f\left(MoP_1, MoP_2, \ldots MoP_m\right) \quad \forall i = 1 : m \tag{1}$$

$$FoM = f\left(MoE_1, MoE_2, \ldots MoE_n\right). \tag{2}$$

The MoP–MoE mapping is first generated through large-scale simulation as it captures all of the complex interactions with high fidelity. Then metamodels of the large-scale simulation are constructed in order to perform evaluations of this mapping in real-time. To aggregate multiple MoEs or multiple simulation outputs into a singled-valued FoM number, this paper provides a preference modeling approach for MoE–FoM mapping. Figure 2 presents a schematic of the entire MoP–FoM mapping process. This mapping is designed with the objective of measuring the impact of component MoP deviations on the overall system effectiveness (FoM) while supporting real-time analysis environments. This mapping process also enables rapid identification of the key parameters controlling a system's behavior while providing a rapid means to perform sensitivity analysis, analysis of alternatives (AoA) and fast optimization.

Section 2 discusses problem formulation. Section 3 discusses a methodology for MoP–MoE mapping while Section 4 discusses a procedure for MoE–FoM mapping. Section 5 presents a case study demonstrating how to apply the entire method through the use of a large scale simulation and discusses how the error propagation from MoP to FoM can be assessed. Section 6 highlights conclusions learned from this work and also identifies the key problems that need to be addressed to advance the field further.

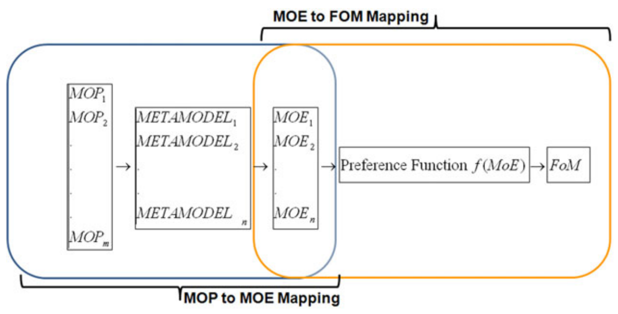

Figure 2. Overview of the MoP–MoE–FoM mapping process with simulation metamodeling for characterization of MoP–MoE mappings and preference modeling for characterization of the MoE–FoM mapping.

2. PROBLEM FORMULATION

The MoP–FoM mapping problem breaks down into two distinct problem phases: MoP–MoE mapping and MoE–FoM mapping. These two components are discussed below.

2.1. The MoP–MoE Problem

The formulation of the MoP–MoE problem centers on the use of a large-scale simulation. This paper develops the MoP–MoE mapping through simulation metamodeling to obtain mathematical approximations that will alleviate the long run times stemming from large-scale simulations. This allows for real-time analysis on lightweight processors, but at the cost of a measurable amount of fidelity.

While a good amount of literature focuses on specific metamodeling techniques as noted above, there exists a need to build a structured approach that is usable by a general analyst. To get there, procedures for selecting the best metamodel family, choosing a robust experimental design, and a validation approach must be developed in a manner that is easily implementable by a general systems analyst. This work systematically works out the steps for these problems, and also identifying critical areas where further research is needed.

2.2. The MoE–FoM Problem

The FoM takes into account the level of preference of a decision maker towards a set of MoE values. Therefore, the MoE–FoM mapping can be pursued by applying a preference modeling approach. multi-attribute utility (MAU) theory [Von Neumann and Morgenstern, 1947] has merits to solve this problem. However, with MAU theory it is often difficult for a decision maker to determine preference for performance measures that are nonmonetary in nature, which does not represent most of the types of systems encountered in systems engineering design. In addition, the amount of uncertainty and level of risk aversion is at times too small to warrant a Von Neumann and Morgenstern approach [Keeney and Winterfeldt, 2007].

This work implements a multi-attribute value (MAV) function [Dyer and Sarin, 1979] approach derived as an approximation of a Von Neumann Morgenstern MAU function [Rosen et al. 2006]. It is an exact characterization of a MAU model when the simulation outputs are normally distributed and captures the decision-maker's preference with regard to uncertainty. Utility functions pertain to preferences for uncertain decision alternatives and value functions relate to a quantitative metric defining the importance of a known outcome level. The MAV function applied here is a cardinal value function that captures the strength of preference between two different MoE levels; detailed explanation into different value function types is described in Chapter 3 of Garvey [2008]. This MAV function approach enables an effective assessment for nonmonetary performances measures; more direct and simplified assessment questions yield better results than a traditional MAU model even though it is essentially a mathematical approximation of it.

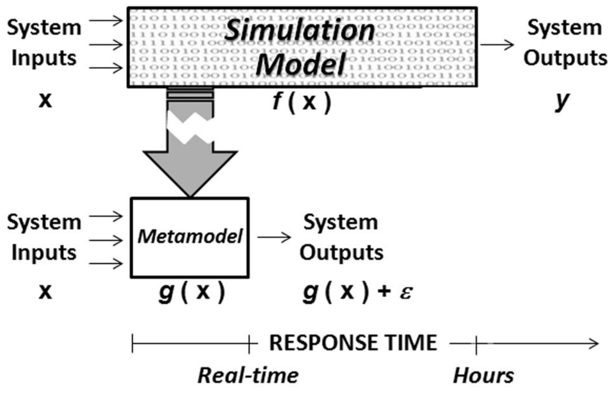

Figure 3. Illustration of simulation and metamodeling with response times.

The MAV value function approach in this paper assumes that the uncertainty in the MoE values is strictly due to intrinsic variability in the simulation model and that there is no fuzziness or ambiguity in the MoEs. In systems where MoEs contain uncertainty or fuzziness due to linguistic measures or information ambiguity, Fuzzy evaluation methods can be applied in a similar fashion to perform the MoE to FoM mapping. Examples of systems where Fuzzy evaluation methods would be more appropriate would include Human systems and Spatial Environmental systems. Bellman and Zedah [1970] provides the first basic structure for a Fuzzy evaluation method and discussions of alternate approaches can be found in Munda, Nijkamp, and Rietveld [1995].

3. MOP–MOE MAPPING PROCEDURE

The basis of the MoP–MoE mapping is considered a derivative of the model-based system engineering approach. The foundation for deriving MoP–MoE mappings is the application of simulation metamodeling to capture the input / output relationships of the validated simulation model. A simulation metamodel [Barton and Meckesheimer, 2006] is constructed from sampling the simulation model at discrete points in the design space. It can be configured as a closed form mathematical expression, but can also be inclusive of rule-based artificial intelligence (AI) approaches.

As shown in Figure 3 metamodels are to be viewed as an alternate analysis means to using the simulation for faster analysis capabilities and for replacing simulations that are essentially unusable because their run times are too long to measure system MoEs. The broken arrow in Figure 3 represents the untethering of a metamodel from a simulation for stand-alone analysis. This occurs after obtaining confidence in its fidelity and satisfactory goodness-of-fit over what is required for its intended use. The metamodel can then be used independently to map the relationships between a subset of system MoPs and a single system MoE.

A metamodel is best understood as a mathematical function with the relationship $y = f(\mathrm{x}) = g(\mathrm{x}) + \varepsilon$ such that y, x are a scalar output and vector valued inputs to the simulation model, respectively. Consider $f(\mathrm{x})$ to be an implicit function representing the mapping between input parameters (MoPs) of the simulation model and the output performance measures or MoEs of the simulation. The simulation metamodel $g(\mathrm{x})$ is an approximation of $f(\mathrm{x})$ with some error term ε which provides confidence in the appropriateness of the metamodel. Figure 4 is a delineation of what defines a metamodeling approach from the metamodeling family (column 1), the fitting strategy (column 2) and the error estimation procedure (column 3)—the terminologies in this figure are widely used in literature on statistics, especially on model selection [Hastie, Tibshirani, and Friedman, 2008; Tibshirani, 1996].

The first step in the metamodeling process is to determine the most appropriate metamodel technique to apply. A best-in-breed procedure is established here that iteratively increases the number of simulation sample points. This significantly reduces the burden of identifying a suitable metamodel technique as well the number of time consuming simulation runs.

The metamodeling family selection tree in Figure 5 provides a holistic method for classification and comparison of metamodeling techniques by showing branches based on simulation characteristics. Metamodeling techniques in the tree are classified by the attributes of the simulation model, namely, number of input parameters, characteristics of the input parameters (such as discrete versus continuous), and complexity of response surface including its noisiness/uncertainties. These classifications were generated from lessons learned through our case studies of metamodeling (discussed in Section 5) as well as through other applications found in the literature. This metamodeling tree guides the user to calibrate the best-in-breed metamodeling technique.

The tree flows from the top down with branching at the first level pertaining to the number of input parameters. The reasoning for the top branch being based on the number of input parameters was built on simulation metamodeling case studies involving two engineering design systems: the first system contained five MoPs or input parameters and the second system contained 44 MoPs. The scenario in the five MoP case is related to a nonaviation checkpoint system and the scenario in the 44 MoP case pertains to a wide-area surveillance/screening. These case studies revealed that Neural Networks' fit was much superior in the case involving 44 parameters. In the five MoP systems, the metamodel error was quite similar for both response surface and neural networks. When error is similar, response surface models are recommended over neural networks since they are more easily implemented, manipulated, utilized and distributed. It should be noted that in these case studies the response surface model was a second-order regression model with a traditional least-squares estimation procedure for estimating the weight coefficients in the model. The neural network fitting approach in this study consisted of using a feed-forward network, with one hidden layer and solving for the weights using back-propagation.

The right-hand side of the tree pertains to the discrete nature of the input parameters. Genetic programs have potential structural advantages over neural networks when the metamodel input space is discrete. Further studies are warranted to exactly extract out the system characteristics that make genetic programs advantageous to neural networks.

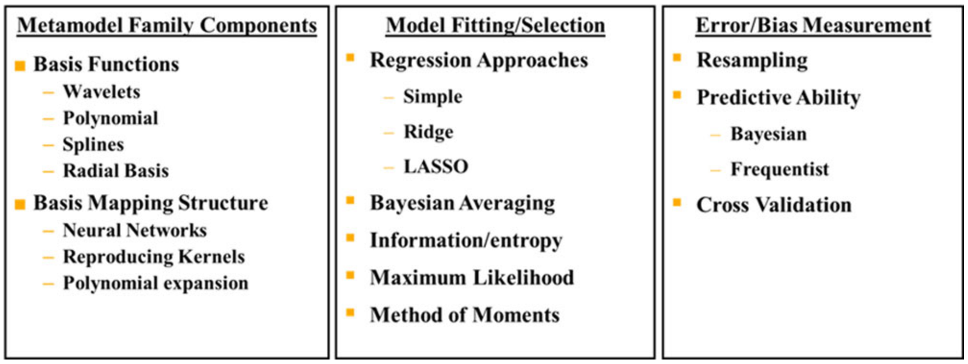

Figure 4. Three key elements of a metamodeling approach include the metamodel family components, model fitting/selection and error/bias measurement: metamodel family components consist of the type of basis function and basis mapping structure, model fitting/selection consists of the technique used to achieve the metamodel with the best goodness of fit and error/bias measurement consists of the type of error and bias statistic used to measure the goodness of fit of the metamodel.

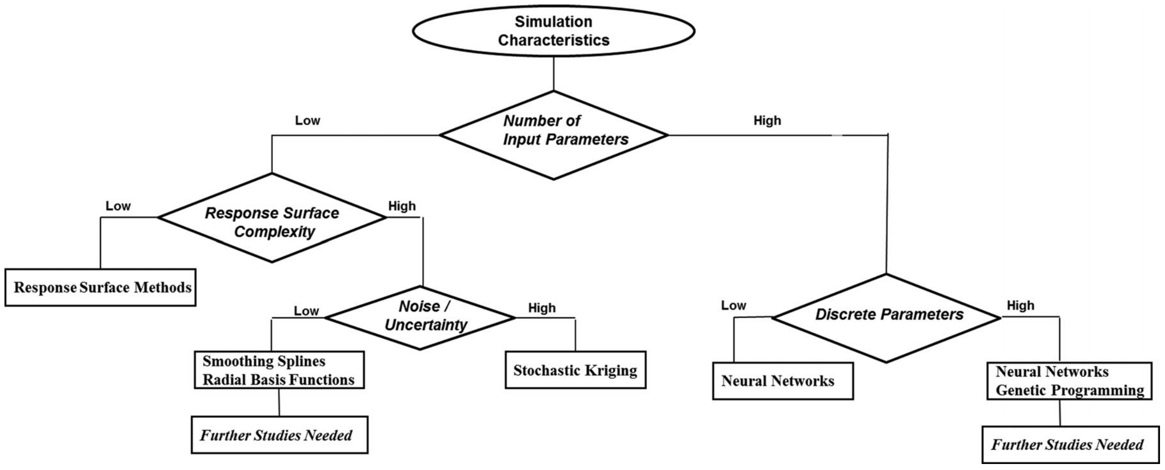

Figure 5. Proposed metamodel families based on simulation characteristics.

Therefore, both methods are recommended in the case of a large number of parameters, which are discrete. In all other cases with large number of input parameters neural networks is the best-suited method. Neural Networks have been shown to perform quite well on complex surfaces [Fonseca, Navaresse, and Moynihan, 2003] with large number of input parameters.

On the left-hand side of Figure 5 the next level down of the tree deals with response surface complexity. Smoothing splines [Eubank, 1988], radial basis functions [Hussain et al., 2002], and Stochastic Kriging [Ankenman et al., 2010] have been shown to work better than traditional response surface models for more complex response surfaces (greater than a second-order polynomial). Stochastic Kriging is more tailored for stochastic simulations and has the ability to fit a very rugged response surface better than neural networks, but the fitting process is very computationally intensive and it thus cannot handle a large number of input parameters. If the function is very deterministic, the splines and radial basis functions methods both warrant inclusion in the best in breed approach and receive the edge over traditional Kriging due to having a smaller risk of miscalibration during implementation [Kleijnen, 2009]. The exact simulation characteristics for applying smoothing splines or radial basis functions require further investigation.

The overall approach we provide for metamodeling integrates the metamodeling family selection tree with cross-validation [Myers et al., 2009] in conjunction with a bootstrapping procedure developed for simulation metamodeling [Kleijnen and Delandre, 2006]. The basic premise is to fit multiple metamodel families simultaneously in order to select the one with the best fit. The reason is that application of the tree in Figure 5 will result in multiple families in certain situations, especially when all the simulation characteristics cannot be ascertained. The sample size is iteratively increased until the desired error is satisfied by one of the metamodel families being tested. Table I contains a step-by-step outline for this generic metamodeling procedure.

The general iterative procedure is to fit the metamodels, assess their goodness of fit via root mean squared error (RMSE) between the simulation model outputs $y_{1,k}$ and metamodel outputs $y_{2,k}$ as shown in Eq. (3). We reserve further discussion on bootstrapping until the MoE–FoM procedure is introduced in Section 4, but it is applied here

Table I. Generalized Approach for Simulation Metamodeling

Initialization
1. Assign each metamodel family to be calibrated = M_j
2. Assign the # of design points to execute the simulation = P
3. Assign the initial number of replications per design point = N
3. Assign the number of design points for validation = k
4. Assign a sample size multiplier = C
5. Assign the desired RMSE goodness of fit = R
5. Set the step indicator $i = 0$
Step
6. $P = P + C^*i$
7. $i = i + 1$
8. Generate a disciplined Experimental Design D having P points
Execute
8. Run N replications of the simulation at each point in D
9. Generate 1000 bootstrap samples of the response across D
10. Using each bootstrap sample:
a. Calibrate M_j for each metamodel family j
b. Compute the bootstrap standard error for family j
11. Measure the mean RMSE for each metamodel R_j
Check
12. If $R_j > R$ for all j then go to step 6
13. Select the metamodel with the best goodness of fit
14. Obtain bootstrap confidence interval for selected metamodel

in order to obtain a standard error estimate and confidence interval for the best fitting metamodel.

$$RMSE = \sqrt{\frac{\sum_{p=1}^{k} \left(y_{1,p} - y_{2,p}\right)^2}{k}}. \tag{3}$$

If RMSE is not at an acceptable level in accordance to the system analyst, then one can proceed to an augmentation of the experimental design, run more simulations, and refit the metamodels; this loop can repeat until the maximum allowable number of simulation runs has been performed or the desired RMSE has been achieved. This procedure also gives the analyst the flexibility to predetermine a ceiling on the number of simulation runs to be performed. This procedure in Table I leaves out mentioning to the type of experimental design to be applied to fitting the metamodels. The strategy for generating an experimental design supporting this procedure is described in the next section.

3.1. Generalized Experimental Design for Metamodeling

Due to their applicability in most cases and mutual complementarity, this experimental design is derived specifically for both response surface models and Neural Networks, but it is applicable to other metamodeling approaches and the best in breed metamodeling approach presented above. Response surface models are more appropriate when there are a small number of input parameters or MoPs, while Neural Networks have the capability to handle a larger set of MoPs for metamodeling.

The experimental design pertains to a hybrid of a Latin hypercube design (LHD) and a factorial design. This design consists of corner points of the design space generated by a factorial or fractional factorial design and a randomized sampling of points on the interior of the design space via a LHD. This design structure is easily augmentable to include extra design points if the initial metamodeling results are not satisfactory. There are other experimental design methods applicable for metamodeling such as Box-Behnken designs, central composite designs, and Placket-Burman designs [Wang and Shan, 2006; Kleijnen, 2004, 2005]. These methods are notably more suitable for simpler response surfaces. However, the hybrid method considered by this paper is, in general, effective in fitting multiple metamodels including response surface models, neural networks, and Stochastic Kriging due to the sampling that occurs at both the edge of the design space and the interior of the design space. The interior sampling also allows for handling more complex response surfaces.

The Latin hypercube design component for N runs and m MoPs will be denoted as an $N \, x \, m$ design. In this design type, each factor appears only once with each of the N equally spaced levels. The location of the design point within the space defined by each level is selected randomly, as random sampling is a common method for generating calibration data in Neural Network metamodels. Due to this random generation of points, significant gaps in the design space are possible. In order to alleviate this problem we apply the following resampling procedure for the LHD component of this hybrid design, which is a modification of a method introduced by Alam, McNaught, and Ringrose [2002]:

1. Create N strata through assigning N equally spaced levels through each design range of each of the m MoP.
2. Randomly sample once from each of the N strata in all m MoPs.
3. Calculate the distances d_{ij} between all design point vectors x in the design space where $d_{ij} = |x_i - x_j| \; \forall i \neq j$.
4. Compute the standard deviation of the distances $S_d = \sqrt{\frac{\sum_{l=1}^{mn} \left(d_{ij} - \bar{d}\right)^2}{\mathrm{N}m(m-1)/2}}$.
5. Repeat steps 1–4 p times.
6. Select the design out of the p randomized LHD designs that minimizes S_d.

The motivation behind the function presented in step six is to provide an automated way to select a randomized design that is not overly clustered in particular areas of the design space. A design with a high S_d would occur if design points are clustered together along with large gaps in the design space, and that should be avoided.

The next step is to integrate the LHD component into the overall hybrid design. We show six levels of granularity of the hybrid design as shown in Table II.

The granularity levels reflect the number of simulation runs to be executed to perform the design. The more complicated the response surface of the simulation is perceived to be, the higher is the design granularity. Moreover, if the user decided

Table II. Granularity Levels for the Generalized Experimental Design

Granularity Level	LHD Component	Factorial component
1	None	2m-2 (Fractional Factorial)
2	m × m runs	None
3	2m × 2m runs	None
4	None	2m (Full Factorial)
5	2m × 2m runs	2m-2 (Fractional Factorial)
6	4m × 4m runs	2m (Full Factorial)

on only a response surface metamodel then the designs with no LHD component can be sufficient (Level 1 and 4). Likewise, if a Neural Network is predetermined as better fitting then removing the factorial component may be appropriate.

4. MOE–FOM MAPPING PROCEDURE

With the first phase of the MoP–MoE–FoM mapping procedure producing metamodels for each MoE, the second phase involves the aggregation of multiple MoE values into a single FoM. This will provide an efficient means for comparing two nondominated candidate solutions and moreover, an automated means for performing optimization of overall system effectiveness.

As shown in Figure 6, there are $2n$ metamodels if the analyst pursues a mapping for the mean and standard deviation of each MoE. Metamodels of standard deviations require observing multiple replications of the simulation model at each design point and are of, course, not relevant in cases of deterministic simulations. The preference modeling approach contained in this section consolidates these metamodel outputs into a single-valued Figure of Merit by first developing component value functions v_i, which map the levels of an individual MoE expected value or standard deviation i to a level of preference scaled between 0 and 1. Then, through assigning weights w_i to these component value functions, we can then develop a MAV function V, which is composed of these component value functions acting as basis functions.

4.1. Form of the MAV Function

A multilinear value model is used to map levels of MoE expected value and standard deviation on [0,1]. This is the form of the value model [Rosen et al., 2006], which is derived as an approximation to the MAV model.

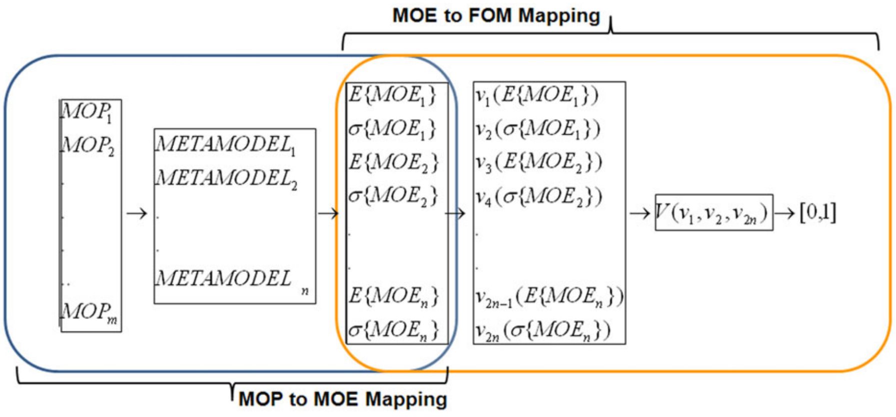

Figure 6. Detailed schematic of the MoP–MoE–FoM process.

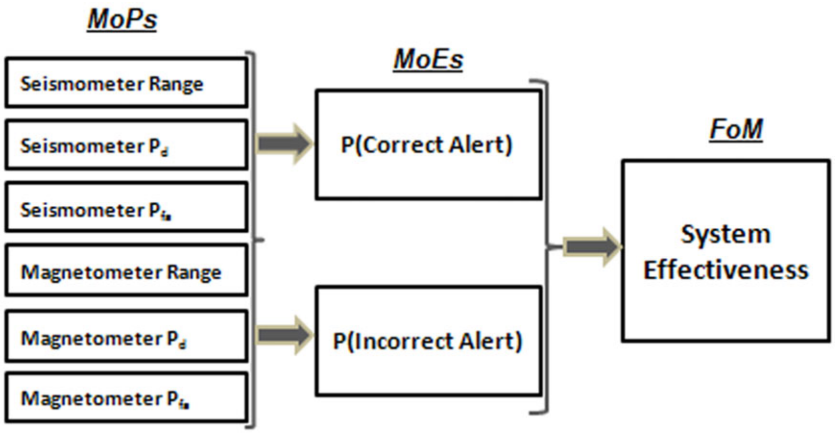

Figure 7. MoP-MoE-FoM mapping system for the case study.

$$V(y) = \sum_{i=1}^{2n} w_i v_i(y_i) + \sum_{i=1}^{2n}\sum_{j>1}^{2n} w_{ij} v_i(y_i) v_j(y_j)$$
$$+ \sum_{i=1}^{2n}\sum_{j>i}^{2n}\sum_{l>j}^{2n} w_{ijl} v_i(y_i) v_j(y_j) v_l(y_l) + \dots$$
$$\dots + w_{123\dots 2n} v_1(y_1) v_2(y_2) \dots v_{2n}(y_{2n}) + \varepsilon. \quad (4)$$

The variables y_i can represent levels of either expected value or standard deviation. Standard deviation terms in the MAV model will incorporate the decision-maker's preference towards risk and uncertainty for that particular MoE. The basis functions $v_i(y_i)$ represent a component value function for a specific MoE expected value or standard deviation. The coefficients w_i can be interpreted as weights representing the relative change in FoM value to a change in a MoE expected value or standard deviation. The error term represents the difference between the MAV model output from Eq. (4) and the human decision-maker's true preference value.

4.2. MAV Function Assessment Procedure

The procedure for generating this preference function is summarized below; the detailed procedure is described elsewhere [Rosen et al., 2006].

1. Implement a series of tradeoff questions to assess weights for each basis value function as well as interaction effects between value functions to be used in the MAV function.
2. Individual value functions are calibrated for both the expected value and standard deviation of each MoE. This will incorporate uncertainty and the decision-maker's preference toward risk into the MoP–MoE mapping.
3. Consolidate the results in I and II into a closed form mathematical expression.

Regarding step II: the assessment of MoE value functions for expected value is achieved through applying the Equisection method of Torgerson [1958]. Individual value functions for standard deviation are calibrated by repeatedly assessing trade-offs between expected values and standard deviations across different paired combinations for the MoEs. Then, the sample value function response for the standard deviation sample points used in the expected value tradeoff questions is computed through the following manipulation of Eq. (4)

$$v(s_i) = \frac{w_a[1 - v(a)]}{w + w_{as} v(a)} \quad (5)$$

where w_a is the weight coefficient for the basis function associated with the given expected value a, and w_{as} is the interaction weight coefficient for the expected value a and the sample standard deviation point $v_i(s_i)$. This equation results by implementing Eq. (4), setting $V(a,b) = V(e,c)$ and solving for $v_i(s_i)$.

After all sample points s_i have been assessed, the method of least squares can be applied to fit a value function through these six sample points $[s_i, v(s_i)]$. After this step, all necessary components of the MoE to FoM model have been calibrated and can be included as input in Eq. (4).

4.3. Error Propagation

There are two types of errors present when fitting simulation metamodels and these errors propagate upward to the FoM level. The first is due to a sampling error from the simulation model. The second is due to bias in the metamodel [Alam et al., 2002] with respect to the simulation model. For instance, if we rerun a simulation over and over again, a biased meta-modeling approach will on average not equal the true underlying simulation model.

There are two sources of bias to be concerned about. The first is when the true underlying simulation model is not contained within the class of possible meta-models. This can arise when there is a misspecification of the meta-models or misspecification to facilitate the estimation approach, but this is greatly alleviated by our best-in-breed approach. The second source of bias occurs when the estimation approach is biased, such as with a Bayesian method.

Metamodeling error due to sampling error can be accounted for through techniques, such as, sub-sampling, jackknifing, and bootstrapping. We perform a bootstrap approach, described below, to measure the propagated error from a set of MoP-MoE metamodels through the FoM.

Let $MoE_i = \{MoE_{i1}, MoE_{i2}, \dots, MoE_{in}\}$ for $i = 1, 2, \dots, I$ be the sample of MoEs at the i^{th} design point of MoPs. The j^{th} bootstrap set of samples is drawn through the following steps:

I. For *i* in 1 to *I:*
   a. Sample (with replacement) from the simulation response of MoE_i
   b. Label this sample of *n points* MoE_i^{*j}

II. Perform the full metamodel fitting procedure on $MoE_1^{*j}, MoE_2^{*j}, \dots MoE_I^{*j}$ to get $MoE^{(*j)}$

III. Repeat steps *I and II a large number of times to achieve M* bootstrap samples

IV. Repeat steps I-III for each metamodel and input each obtained $MoE^{(*j)}$ to achieve $FoM^{(j)}$

The bootstrap standard error (StdErr) estimate then becomes:

$$StdErr = \sqrt{(m-1)^{-1} \sum_{j=1}^{m} \left(FoM^j - mean(FoM^j)\right)^2} \quad (6)$$

5. CASE STUDY ON COMPLETE MOP-FOM MAPPING PROCEDURE

The case study involves a wide-area surveillance/screening scenario. Figure 7 provides a schematic representation of the key components and parameters surrounding the assessment of an area wide probability of detection (P_d) for pedestrians along with P_d for vehicles.

The system in Figure 7 involves screening technologies, where seismometers detect vibration due to people's motion

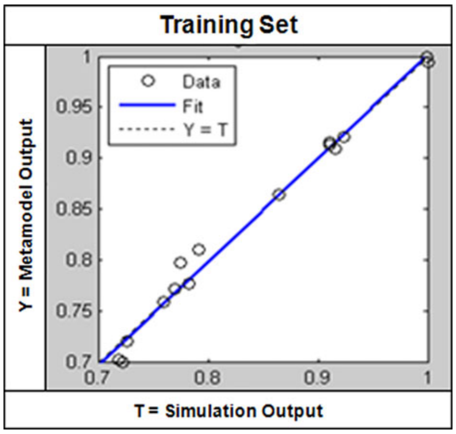

Figure 8. Fit of pedestrian P_d metamodel (training data).

and magnetometers detect vehicles. The MoPs in this example represent performance measures for the system technology components involving seismometers and magnetometers. The MoEs in the example are the probability of a correct alert and the probability of an incorrect alert, which are system-level measures and collectively define the overall system effectiveness or Figure of Merit.

The analysis consists of understanding the relationship between key decision variables pertaining to an array of seismometers and magnetometers that are to be allocated across an area to detect pedestrians and vehicles. Four decision variables were selected for this case study based on the modeling capabilities of the large-scale simulation being utilized:

- x_1 = Number of seismometers
- x_2 = Seismometer Location
- x_3 = Range of the seismometer
- x_4 = Location of magnetometers

The effect of radars is also considered in this area wide P_d assessment. Radars are fixed and represent a baseline for P_d. The assessment involves the additional improvement to P_d given by the seismometers and magnetometers. The items of interest that the sensors are attempting to detect traverse along discrete paths through the area. Thus the simulation measures the proportion of detections made on each path and averages them to obtain a system-level $P_{d.}$

5.1. MoP–MoE Calibration

This study involves the following MoEs with the following output ranges:

- Pedestrian P_d
    - Expected value [0–1]
    - Standard deviation [0–0.25]
- Vehicle P_d expected value [0–1]

The study started with an agent-based simulation that provided outputs of the three MoEs provided above. Vehicle detection P_d had an insignificant variance so that was not included in the MoP-FoM mapping. This model took over six hours to run on a standard desktop personal computer; this illustrates the need for real-time analysis. Given the large amount of run time required, only forty-five design points were able to be experimented in total, with ten replications at each design point. Twenty-one design points were allocated to calibration of the pedestrian P_d metamodel and twenty-four were allocated to the vehicle P_d metamodel.

The pedestrian P_d metamodel was best fit through a Neural Network. We present the goodness-of-fit through regression plots on the training, validation, and testing data. Figure 8 presents a scatter plot of metamodel outputs, versus simulation outputs for the data set that was used to calibrate the model. If a perfect fit is achieved, all points would fall on the diagonal line extending from the origin. Overall, the P_d metamodel was able to achieve an R value (linear correlation coefficient) of over 0.992 with the simulation model. An RMSE of 0.0178 was achieved between the metamodel and simulation outputs on the training data.

The validation partition of the data is used for model selection and is used during the training phase of the Neural Network to check for overfitting. If the accuracy over the validation data stays the same or decreases, then it can be concluded that overfitting is beginning to occur and training

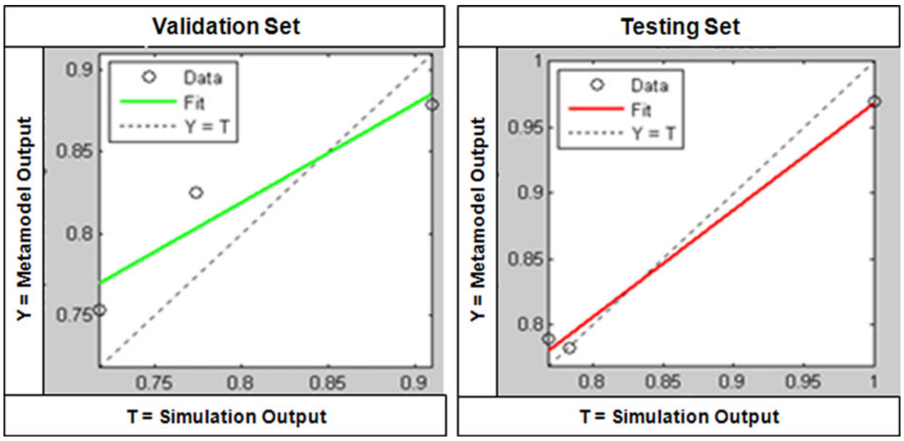

Figure 9. Fit of pedestrian P_d metamodel (validation and testing).

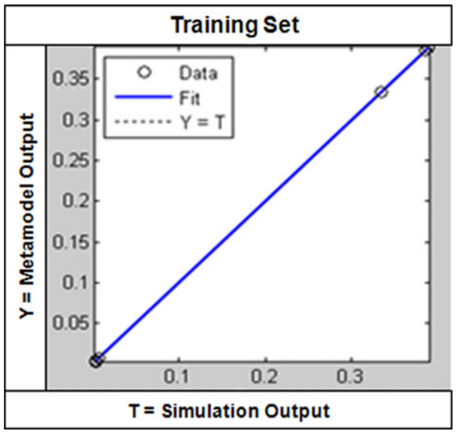

Figure 10. Fit of vehicle P_d metamodel (training data).

should stop. The testing partition of the data is used to evaluate how well the Neural Network can performed on points not used for training or can represent the simulation under the simulation inputs that were not used to fit the metamodel. Figure 9 presents the metamodel, versus simulation output regression plots for the validation and testing data. The R value for the validation and testing data set was 0.950 and 0.963, respectively. An RMSE of 0.0402 was achieved on validation partition of the data, and an RMSE of 0.0213 was achieved for the testing partition of the data.

For the pedestrian P_d standard deviation metamodel a Response Surface Metamodel was able to achieve a good fit. The following functional form yielded $R^2 = 0.986$

$$g(x) = 0.095 + 0.0001x_1 - 0.0003x_2 + 0.0002x_3 + 0.0012x_4 \\ -0.00002x_1x_2 + 0.0004x_1x_4 + 0.000065x_2x_4 \qquad (7)$$

Next, a Neural Network metamodel was calibrated for the vehicle P_d expected value. Figure 10 shows the metamodel versus simulation output regression plots for the training data. Overall, the metamodel for percentage of vehicle detection achieved an R value of 0.999, meaning nearly a perfect fit. There was also a cluster of points close to zero, making this a difficult metamodel to calibrate; this could be done via a Neural Network structure. The RMSE value on the training data was 0.000032.

Figure 11 presents the metamodel versus simulation output regression plots for the validation data and testing data. The R values for validation and testing were 0.976 and 0.986, respectively. The RMSE values, respectively, for the validation and testing partition of the experimental design were 0.041 and 0.057.

The RMSE value provides the average error in the metamodel fit across the entire design space. There are other methods that can lend further assistance. Computing a PRESS statistic (Predicted Residual Sum of Squares) to measure bias can support metamodel selection or misspecification of metamodels, and a bootstrap calculation of standard error (discussed in Sec. 4.3) can provide information on the variability in the metamodel fit due to the intrinsic variability inside the simulation model. Generally speaking, PRESS statistic and bootstrap error estimates require strong computational

Table III. Rating Questions for MAV Function Weight Assignment

Design Point	E (Seismometer P_d)	St.Dev. (Seismometer P_d)	E (Vehicle P_d)	Rating
A	0	0	0	10
B	1	0	0	65
C	0	0.25	0	0
D	1	0.25	0	55
E	0.5	0.125	0.5	50
F	0	0	1	53
G	1	0	1	100
H	0	0.25	1	55
I	1	0.25	1	95

resources, such as parallel processors, and this is being implemented in future work for metamodeling through an integrated cloud-computing architecture.

5.2. MoE-FoM Calibration

The MoE to FoM mapping procedure is applied to all three metamodels that have been constructed:

- Pedestrian P_d expected value metamodel [0–1]
- Pedestrian P_d standard deviation metamodel [0–0.25]
- Vehicle detection P_d expected value metamodel [0–1]

To generate the MoE–FoM mapping, a system subject matter expert (SME) is utilized to answer the tradeoff questions regarding system MoEs that are necessary to calibrate a MoE to FoM preference function. Within the MAV function three value functions are included pertaining to the three metamodels that have been constructed as noted above. The first step is to assess the weights w of each basis function with the grand MAV function (Eq. (4)). The SME must make the following *holistic ratings* on a scale from 0 to 100 involving MoE expected values and standard deviations at the edge of their preference space. Table III provides the results of these ratings; note that the design points, A through I, are not an ordering that is based on a ranking.

To infer these ratings from the SME (subject matter expert), the rating questions are posed in the following manner: Rate each design point on a scale of 0–100 with a score of 100 representing most preferred solution and a score of 0 representing least preferred solution. The resulting ratings are provided in the right-hand column of Table III. From these ratings the weight coefficients to the MAV function are calibrated through the method of least squares. To do this the MoE design points are transposed back to value function levels of 0 and 1 that they were initially assigned to represent.

The approximated weights obtained through the method of least squares are $[56.8, 33.0, 44.75, -14.0, -11.5, 34.0]$. For this example, the MAV function with the newly assessed main effect w_i and interaction weights w_{ij}:

$$V(y) = 8.67 + 56.8v_1(y_1) - 33.0v_2(y_2) + 44.75v_3(y_3) \\ - 14.0v_1(y_1)v_2(y_2) - 11.5v_1(y_1)v_3(y_3) \\ + 34.0v_2(y_2)v_3(y_3) \qquad (8)$$

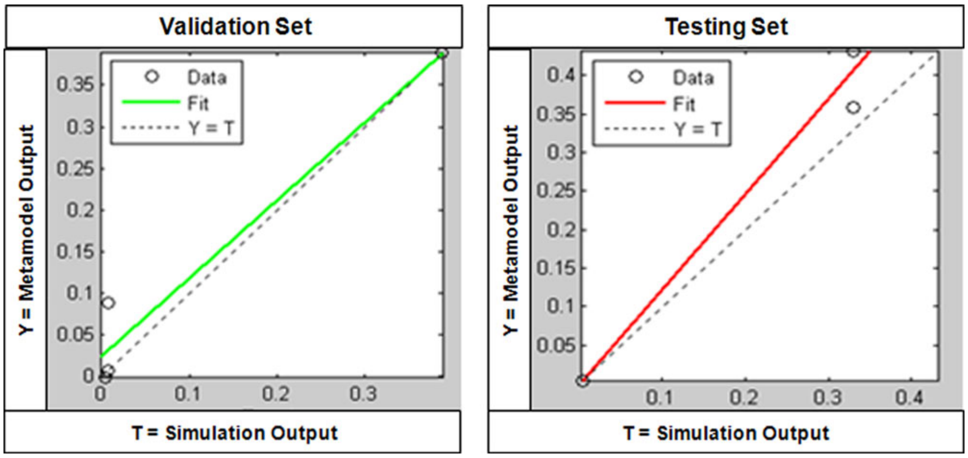

Figure 11. Fit of vehicle P_d metamodel (validation and testing).

Table IV. Resulting Sample Points from the Equisection Method

E (Pedestrian P_d)	$v(y)$	E (Vehicle P_d)	$v(y)$
0	0	0	0
0.6	0.125	0.05	0.125
0.7	0.25	0.1	0.25
0.8	0.375	0.15	0.375
0.85	0.5	0.19	0.5
0.9	0.625	0.23	0.625
0.94	0.75	0.27	0.75
0.98	0.875	0.3	0.875
1	1	1	1

The next step in calibrating this MAV function is to calibrate the individual basis functions $v(y)$ within Eq. (4). We begin by assessing the pedestrian P_d expected value and vehicle detection P_d expected value via the Equisection Method. These component value functions relate to $v_1(y_1)$ and $v_3(y_3)$ in Eq. (8) above. Table IV above represents the sample points that need to be assessed through the Equisection Method to fit these value functions.

The SME is then instructed to provide a level of expected value x such that an increase in the MoE from a to x is equally preferable to x to b. The process begins by using the endpoints to assess the MoE level for $v(y) = 1$ and repeats by assessing midpoints until all points are assessed.

The method of least squares is applied to fit the value functions for pedestrian P_d expected value and vehicle detection P_d expected value. The two functions below represent the fitted value functions for these two MoEs pertaining to MoE expected value. The resulting value functions for the pedestrian detection P_d expected value and vehicle detection P_d expected value, respectively, are:

$$v_1(y_1) = 0.0065 - 1.10y_1 - 2.03y_1^2 \qquad (9)$$

$$v_3(y_3) = -0.05 + 3.61y_3 - 2.55y_3^2 \qquad (10)$$

Table V. Trade-off Questions for Standard Deviation Value Function

E(P_d), St.Dev. (P_d)	E(P_d), St.Dev. (P_d)
(1, 0.25)	(y, 0.2)
(1, 0.25)	(y, 0.15)
(1, 0.25)	(y, 0.1)
(1, 0.25)	(y, 0.05)
(1, 0.25)	(y, 0)

Table VI. Sample Points for Standard Deviation Value Function

E(P_d), St.Dev.(P_d)	E(P_d), St.Dev.(P_d)
(1, 0.25)	(0.95, 0.2)
(1, 0.25)	(0.9, 0.15)
(1, 0.25)	(0.8, 0.1)
(1, 0.25)	(0.7, 0.05)
(1, 0.25)	(0.5, 0)

The final step is to calibrate the component value function pertaining to pedestrian P_d standard deviation. In the assessment procedure, the decision maker is presented these questions via Table V. The decision makers must assess the level of expected value $(1 - y)$ they are willing to sacrifice to improve the standard deviation to the level noted in the right column of Table V.

The SME is instructed here to provide a level of expected value y that makes the pair of expected value/standard deviation MoE values equally preferable. Table VI provides the resulting expected values provided by the SME:

The following y, $v(y)$ sample points for standard deviation are obtained through Eq. (5): $y = [0.2, 0.14, 0.1, 0.5, 0]$, $v(y) = [0.45, 0.65, 0.83, 0.95, 1]$. The least squares estimation results in Eq. (11).

$$v_2(y_2) = 1.0046 - 0.74y_2 - 10.29y_2^2 \qquad (11)$$

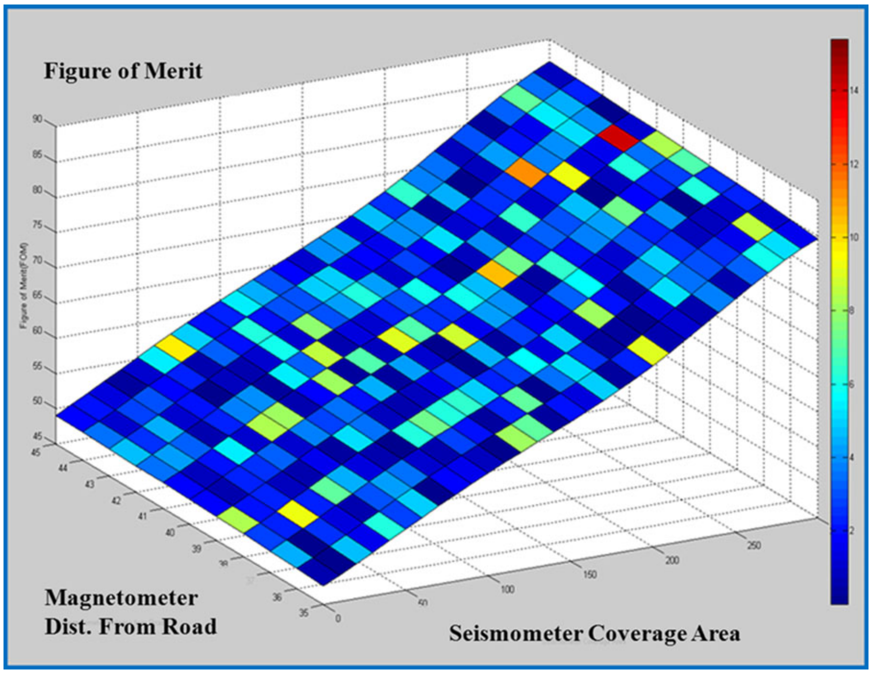

Figure 12. FoM response surface with colors depicting bootstrap confidence bounds; colors are not distinct in a black and white representation. Online version of the article shows colors distinctly.

At this point all components of the value model have been assessed. The MoE to FoM mapping for this study is:

$$\begin{aligned}V(y) = 8.67 + 56.8v_1(y_1) - 33.0v_2(y_2) + 44.75v_3(y_3)\\ -14.0v_1(y_1)v_2(y_2) - 11.5v_1(y_1)v_3(y_3)\\ +34.0v_2(y_2)v_3(y_3)\end{aligned} \quad (12)$$

where: $v_1(y_1) = 0.0065 - 1.10y_1 - 2.03y_1^2$

$$v_2(y_2) = 1.0046 - 0.74y_2 - 10.29y_2^2$$

$$v_3(y_3) = -0.05 + 3.61y_3 - 2.55y_3^2$$

y_1 = Pedestrian Expected Value P_d,
y_2 = Pedestrian P_d Standard Deviation,
y_3 = Vehicle Expected Value P_d

Now that the MoP to FoM mapping has been fully defined, the sensitivities of individual MoPs can be examined to illustrate the merit of this MoP-FoM metamodeling approach. Matlab code was generated to combine the fitted metamodels together with the MAV model to enable 3D Response Surface plots. These plots can be rapidly developed to examine local influences of specified MoPs on the overall FoM.

The plot in Fig. 12 presents a visual depiction of the FoM response surface with respect to the MoPs of magnetometer spacing and seismometer coverage area. Here all other MoPs were set to fixed values. The metamodels fitted above along with the MoE-FoM preference function were utilized to create this FoM response surface under the specified ranges of these MoP values.

The FoM response surface depicted in Figure 12 also conveys the level of propagated error stemming from the metamodels. The color of the patches on the response surface

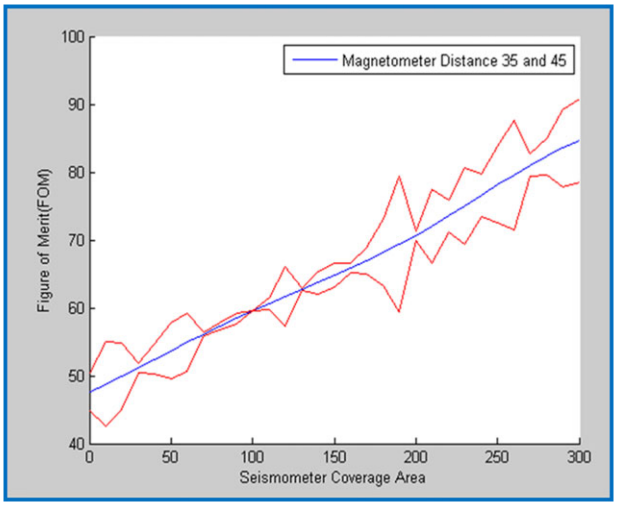

Figure 13. FoM response surface slice with confidence bounds.

represents the size of the 95% confidence interval on the FoM due to metamodel error. Color frequency is related to confidence interval size; dark blue represents a small interval of close to zero and red represents a large interval (+/- 14) on the FoM. The exact values are noted on the color scale on the right side of Figure 12. The confidence interval on the FoM response surface is achieved through executing the bootstrap method on the metamodels to measure the standard error at discrete points along the surface.

An interesting observation is the level of fluctuation in the error on the response surface. The bootstrapping method represents the aggregated variability in fit of the metamodels due to the sampling error of the simulation and Figure 12 depicts how this variability fluctuates across different levels of MoP values. Figure 13 presents a two-dimensional slice of the response surface and it shows error fluctuations.

It should be noted that there were only twenty-one design points (with ten replications each) to fit the pedestrian P_d metamodel and twenty-four points to fit the vehicle P_d metamodel. Having such limited design points can lead to this fluctuation (with sharp spikes) in error. The utility of the visualization of the results in Figures 12 and 13 is that the system analyst can obtain a level of confidence on the FoM for the most preferred system inputs and MoP configurations that he or she decides to implement.

6. CONCLUSIONS

This paper addresses a need for a paradigm shift in Systems Engineering by providing a means for real-time and agile quantitative analysis of complex systems via modest computational resources. A structured approach for simulation metamodeling for MoP to MoE mapping is developed. The intent is tohave a generalizable procedure, which is useable by general systems analyst. In addition, this paper has built a MAV function fitting approach for aggregating multiple MoE values from these metamodels into a single FoM. The two steps together provide a complete procedure for directly mapping MoPs to FoM, and it enables a quantitative and efficient means for performing systems analysis.

A case study has been performed on a large scale agent-based simulation. This large scale simulation required six hours of runtime for ten replications under a single scenario and the metamodeling approaches applied were able to capture a significant amount of the variability contained within that simulation model affected by the MoPs involved in the study. The MoE-FoM mapping was also implemented using a SME to yield a complete MoP-FoM mapping.

The key points for further studies have been identified, and many of these revolve around the simulation metamodeling component. The metamodel selection tree in Figure 6 and the structure of MoE Design of Experiments laid down in Table II can be further investigated to advance metamodeling generalization. Studies into the performance of Radial Basis Functions and Spline functions under different situations and studies on Genetic Programming, versus Neural Networks will enhance the knowledge and help further define this tree. Fully comparing metamodeling methods for this MoP-FoM framework requires additional metrics other than RMSE and R values. As discussed in Fig. 12, bootstrap techniques are employed to estimate the standard error of the metamodel. Bootstrap techniques can be coupled with PRESS statistic to estimate the bias of the metamodel and with an ANOVA-based MSE to estimate expected sample sizes. The difficult aspect of any of these error measures is that the user must relate a quantitative measure to the intended use of the metamodel to infer suitability. This results in some subjective decision making on the analyst's end.

Metamodeling with agent-based simulations poses some challenges, especially from these models representing a large number of behavior states. To include all behavior states as individual variables in the metamodel would require a large experimental design, and this may result in many simulation runs for calibration. Neural Networks seem to handle simulations with many input variables, but factor screening techniques should also be explored in details in this situation. It must be alerted that screening can lead to information loss in the metamodel as factor screening techniques are prone to erroneously screening out important factors and missing factors that are part of critical two-way interaction effects. However, screening out insignificant variables apriori can allow for effective utilization of metamodeling techniques such as Stochastic Kriging [Rosen and Guharay, 2013].

Further research is needed to answer the question of metamodel sufficiency. The sufficiency or the readiness for intended use of the metamodel needs to be tied into the validation statistics and be implemented on the fitted metamodel. Different users require different levels of sufficiency and different levels of goodness-of-fit. This is an issue that needs to be formally delineated and then integrated into the standard metamodeling procedure to better guide the analyst through the metamodeling calibration process.

ACKNOWLEDGMENTS

This work is supported by the Systems Engineering portfolio of the MITRE Innovation Program. The authors are thankful to the reviewers for very constructive and valuable comments and suggestions. The authors are thankful to Ms. Marie Francesca and Dr. Kevin Cabana for many insightful suggestions. Thanks are also due to Tobin Bergen-Hill for his help in executing large-scale M&S experiments.

REFERENCES

F. Alam, K. McNaught, and T. Ringrose, A comparison of experimental designs in the development of a neural network simulation metamodel, Simulat Model Pract Theory 12 (2002), 559–578.

H. Al-Hindi, Approximation of a discrete event stochastic simulation using an evolutionary artificial neural network, Eng Sci 15(1) (2004), 3–16.

B. Ankenman, B. Nelson, and J. Staum, Stochastic kriging for simulation metamodeling, Oper Res 58(2) (2010), 371–382.

R. Barton and M. Meckesheimer, Metamodel-based simulation optimization, in S.G. Henderson (Editor), Handbook in OR & MS, Vol. 13, Elsevier, Amsterdam, Netherlands 2006, 535–574.

R. Bellman and L. Zadeh, Decision-making in a fuzzy environment, Manag Sci 17 (1970), 141–164.

G. Box, and N. Draper, Empirical model-building and response surfaces, John Wiley and Sons, New York, 1987.

L. Chwif and R. Paul, On simulation model complexity, Proceedings of the 2000 Winter Simulation Conference, Orlando, FL, 2000, pp. 449–445.

C. Deboor, A practical guide to splines, Springer-Verlag, New York, 1978.

J. Dyer and R. Sarin, Measurable multiattribute value functions, Oper Res 27(4) (1979), 810–822.

N. Dyn, D. Levin, and S. Rippa, Numerical procedures for surface fitting of scattered data by radial functions, SIAM J Scient Stat Comput 7, 639–659, 1986.

R. Eubank, Spline smoothing and nonparametric regression, Marcel Dekker, New York, 1988.

D. Fonseca, D. Navaresse, and G. Moynihan, Simulation metamodeling through artificial neural networks, Eng Appl Artif Intell 16 (2003), 177–183.

P. Garvey, Analytical methods for risk management: A systems engineering perspective, Chapman and Hall/CRC, London, England, 2008.

Y. Haimes, Modeling complex systems of systems with phantom system models, Syst Eng 15(2), 2012.

A. Hall, III, Metasystems methodology: A new synthesis and unification, Pergamon Press, New York, 1989.

D. Hall and S. McMullen, Mathematical techniques in multisensor data fusion, 2nd edition, Artech House, Boston, 2004.

T. Hastie, R. Tibshirani, and J. Friedman, The elements of statistical learning data mining, inference, and prediction, 2nd edition, Springer, New York, NY, 2008.

B. Horowitz and J. Lambert, Learn as you go systems engineering, IEEE Trans Syst, Man, and Cybernet, A 36(2) (2006), 286–297.

M. Hussain, R. Barton, and S. Joshi, Metamodeling: Radial basis functions, versus polynomials, Eur J Operat Res 138 (2002), 142–154.

International Council on Systems Engineering (INCOSE), Systems engineering handbook version 3.1, August 2007, pp. 3.3 to 3.8.

R. Kenney and D. von Winterfeldt, Practical value models advances: Practical value models, Ch. 13, Cambridge University Press, Cambridge, England, 2007.

J. Kleijnen and W. van Beers, Application-driven sequential designs for simulation experiments: Kriging metamodeling, 55 (2004), 876–883.

J. Kleijnen, An overview of the design and analysis of simulation experiments, Eur J Oper Res 164 (2005), 287–300.

J. Kleijnen, Kriging metamodeling in simulation: A review, Eur J Oper Res 192 (2009), 707–716.

J. Kleijnen and D. Delandre, Validation of regression metamodels in simulation: Bootstrap approach, Eur J Oper Res 170 (2006), 120–131.

J. Kleijnen and R. Sargent, A methodology for fitting and validating metamodels in simulation, Eur J Oper Res 120 (2000), 14–29.

G. Klir, Review of model-based systems engineering, Int J Gen Syst 25(2) (1996), 179–180.

J.R. Koza, Genetic Programming: On the programming of computers by means of natural selection, MIT Press, Boston, 1992.

V. Le, G. Bloch, and F. Lauer, Reduced-size kernel models for nonlinear hybrid system identification, IEEE Trans Neural Networks 22(12) (2001), 2398–2405.

R. Lippman, An introduction to computing with neural nets, IEEE ASSP Mag (1987), 4–22.

L. Ljung, System identification: Theory for the user, 2nd edition, Prentice Hall PTR, Upper Saddle River, New Jersey, 1999.

M. Meghabghab, Iterative radial basis functions neural networks as metamodels of stochastic simulations of the quality of search engines in the World Wide Web, Inform Process Manag 37 (2001), 571–591.

R. Michalski, A theory and methodology of inductive learning, machine learning: An artificial intelligence approach, TIOGA Publishing Co., Palo Alto, 1983.

G. Munda, P. Nijkamp, and P. Rietveld, Qualitative multicriteria methods for fuzzy evaluation problems: An illustration of economic-ecological evaluation, Eur J Oper Res 82 (1995), 79–97.

R. Myers, Response surface methodology, Allyn and Bacon, Boston, 1976.

R. Myers, M. Montgomery, and C. Anderson-Cook, Response surface methodology: Process and product optimization using designed experiments, 3rd edition, Wiley, New York, 2009.

R. Myers, V. Ugru, X. Luo, and R. Grandhi, MIDAS for pre- and post-processing of ASTROS unsteady aerodynamic flutter models, Proceedings of the 6th AIAA/NASA/ISSMO Symposium of Multi-disciplinary Analysis and Optimization, New Orleans, LA, 1996.

J. Ottino, Complex systems, AIChE J 49(2) (2003), 292–299.

C. Osorio, and L. Chong, An efficient simulation-based optimization algorithm for large-scale transportation problems, Proceedings of the 2012 Winter Simulation Conference, Berlin, Germany, 2012.

C. Osorio and H. Bidkhori, Combining metamodel techniques and Bayesian selection procedures to derive computationally efficient simulation-based optimization techniques, Proceedings of the 2012 Winter Simulation Conference, Berlin, Germany, 2012.

S. Rosen, C. Harmonosky, and M. Traband, A simulation optimization method that considers uncertainty and multiple performance measures, Eur J Oper Res 181 (2006), 315–330.

S. Rosen and S. Guharay, A Case Study Examining the Impact of Factor Screening For Neural Network Metamodels, Proceedings of the 2013 Winter Simulation Conference, Washington, D.C., 2013.

S. Rosen, C. Saunders, and S. Guharay, Metamodeling of time series inputs and outputs, Proceedings of the 2012 Winter Simulation Conference, Berlin, Germany, 2012.

J. Sacks, W. Welch, T. Mitchell, and H. Wynn, Design and analysis of computer experiments, Statistical Science 4 (1989), 409–435.

M. Shin, R. Sargent, and A. Goel, Gaussian radial basis functions for simulation metamodeling, Proceedings of the 2002 Winter Simulation Conference, Berlin, Germany, pp. 483–488, 2002.

J. Staum, Better Simulation Metamodeling: The why, what, and how of stochastic kriging. Proceedings of the 2009 Winter Simulation Conference, Austin, TX, 2009, pp. 119–133.

B. Sudret, Meta-models for structural reliability and uncertainty quantification, Fifth Asian-Pacific Symposium on Structural Reliability and its Applications, Singapore, 2012, pp. 1–24.

R. Tibshirani, Regression shrinkage and selection via the Lasso, J R Stat Soc, B: Methodol 58 (1996), 267–288; and references therein.

W. Torgerson, Theory and methods of scaling, Wiley, New York, 1958.

J. Von Neumann, O. Morgenstern, Theory of games and economic behavior, Princeton University Press, Princeton, 1947.

G. Wang, S. Shan, Review of metamodeling techniques in support of engineering design optimization, J Mech Des 129 (2006) 370–380.

W. Wymore, Model-based systems engineering, CRC Press, Inc., Boca Raton, 1993.

Scott L. Rosen is a Group Leader for the Probability and Statistics Group within the MITRE Corporation's Operations Research Department. He has spent the last eight years at MTIRE applying Operations Research to some of the United States' most critical and challenging Systems Engineering problems. His research interests include Simulation Optimization, Simulation Metamodeling, Decision Analysis, and Quantitative Systems Engineering. He received his B.S. from Lehigh University in Industrial and Systems Engineering in 1998 and a M.S. and Ph.D. in Industrial Engineering and Operations Research from Penn State University in 2000 and 2003, respectively.

Christopher P. Saunders is a mathematical statistician with the MITRE Corporation; he also holds an assistant professorship of statistics at South Dakota State University. He received his Ph.D. in statistics from the University of Kentucky before accepting an Intelligence Community postdoctoral fellowship. His current research focus is related to high dimensional pattern recognition and approximate Bayesian inference with applications to time series analysis and forensic identification.

Samar K. Guharay has been leading research in the area of interdisciplinary problems covering novel sensing technology, algorithms, and rapid systems engineering analysis. He received his Ph.D. in Physics in 1980 from the University of Calcutta, India. Dr. Guharay participated in many national and international research programs over more than thirty years of his professional career. He has been affiliated with universities in different capacities. In addition to leading research programs at MITRE he has a long background as PI/Co-PI of projects sponsored by multiple agencies including NSF, NIH, DoE, DoD, and industries.