Proceedings of the 2004 Winter Simulation Conference
R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

USING MORRIS' RANDOMIZED OAT DESIGN AS A FACTOR SCREENING METHOD FOR DEVELOPING SIMULATION METAMODELS

Fasihul M. Alam

Department of Information Systems & Computing
Brunel University
Uxbridge, Middlesex
UB8 3PH, U.K.

Ken R. McNaught
Trevor J. Ringrose

Engineering Systems Department
Cranfield University, RMCS Shrivenham
Swindon, SN6 8LA, U.K.

ABSTRACT

Simulation metamodels have been used for optimization, prediction, sensitivity analysis and understanding of complex, real-world systems. Since most simulation models contain a large number of input parameters, it is of great interest to determine the most important ones to include in a metamodel given a particular modeling context, i.e. given a particular set of questions which are to be addressed by the metamodel. This paper employs Morris' randomized one-at-a-time (OAT) design as a factor screening method prior to developing a number of simulation metamodels. The approach is illustrated with reference to a stochastic combat simulation, called SIMBAT.

1 INTRODUCTION

In computer simulation experiments, an important question is – which factors are really significant when there are potentially a large number factors involved? This question seems rather prominent especially when the ultimate goal of the whole experimentation is to develop metamodels from a simulation. It is unlikely, though, that all of the input parameters of a simulation have a major impact on its response, and generally it is assumed that only a subset of factors will be important (Morris 1987). For a simulation with even a moderately large number of factors, the number of possible factor-level combinations can easily explode far beyond anything remotely practical. To avoid this unnecessary explosion of simulation configurations, it is desirable to detect at an early stage of the experimentation which factors are important and which are not. The unimportant factors are then dropped from consideration by fixing them at some reasonable values, and further experimentation will only be carried out with the more important factors. The nature of the situation under study will often mean that many of the factors can be fixed at particular levels which are defined by the situation. Of the remaining factors, several will usually turn out to be unimportant in this situation. It is the role of a factor screening method to weed out these unimportant factors. For details on metamodeling, see Barton (1992); Kleijnen and Sargent (2000); Hurrion and Birgil (1999); Clarke, Griebsch and Simpson (2003).

We believe this *factor-screening* method is crucial when developing metamodels from a simulation with a large number of input factors and the response surface is complex or not very well-behaved. In the literature, screening designs include random designs, supersaturated designs (require fewer configurations than factors), group-screening designs, and so on (Kleijnen 1987). Several approaches have been proposed in the literature for *factor-screening*, e.g. Morris' (1991) randomized OAT design, the iterated fractional factorial designs (IFFDs) introduced by Andres and Hajas (1993) and the sequential bifurcation (SB) proposed by Bettonvil and Kleijnen (1997). More recently, Trocine and Malone (2001) advocated a screening design employing genetic algorithms. However, in general, relatively little attention has been paid to screening designs in metamodeling research, despite its obvious importance.

Group screening designs, like SB, have been used in simulation, and proved effective for a small to medium number of factors. SB allows a large number of factors to be considered using a relatively small number of simulation configurations. The underlying assumptions of SB are that a first-order polynomial, possibly augmented with two factor interactions, is an adequate function to approximate the simulation response surface, and the response function is either nondecreasing or nonincreasing. SB can provide efficient and accurate results if the above conditions are met. However, such conditions cannot always be assumed. For example, the underlying simulation considered in this paper produces a response surface which is complex by nature, and which does not satisfy the conditions of SB.

Morris' randomized OAT design is another factor screening method that has such flexibility. The advantage of this approach is that it does not make explicit assumptions about the system, e.g. there being relatively few factors that have significant influence on the simulation re-

sponse, or the adequacy of a low-order polynomial as an approximation to the computational model. The number of different simulation configurations required by Morris' method is linear in the number of factors and the results are easily interpreted in a lucid, graphical way. However, the design does not provide estimations for factor-interactions, which is its main limitation.

In this article, we investigate the usability of Morris' OAT design as a potential factor-screening method to be used in developing metamodels from a complex, combat simulation. It is shown that the method is quite effective and is easy to apply. In Section 2, we describe Morris' randomized OAT design as a potential factor screening method for the underlying simulation (largely drawn from Campolongo, Kleijnen and Andres, 2000). Section 3 demonstrates a case study from the underlying simulation, SIMBAT. How Morris' design is applied to the case study and the subsequent results are presented in Section 4. Section 5 concerns the development of metamodels using the important factors derived in Section 4. Finally, Section 6 summarizes the findings and draws conclusions from the article.

2 MORRIS' RANDOMIZED OAT DESIGN

In its original form, Morris' method considers a computationally expensive model, involving a large number of factors, and where the number of simulation runs required is proportional to the number of factors, k. Morris intends to isolate the factors that have either negligible effects or linear and additive effects, or non-linear or interaction effects on the simulation response. Based on individual randomised OAT designs, the method estimates the effect of changes in the level of each factor, and evaluates it in turn.

The basic idea of the method is related to a sample of independently observed *elementary effects*. These are then exploited through a statistical analysis to measure the relative importance of an input (or the sensitivity of the simulation output for a particular input). Suppose, for a simulation model, the k-dimensional factor vector $\mathbf{x}$ has components x_i, each of which can take p discrete values in the set $\{0, 1/(p\text{-}1), 2/(p\text{-}1), \ldots, 1\}$. For simplicity, it is assumed that each x_i is scaled to have a region of interest equal to [0, 1]. The experimental domain, Θ, is then a k-dimensional p-level grid. Let Δ be a predetermined multiple of $1/(p\text{-}1)$. Morris defines the *elementary effect* of the ith factor at a given point $\mathbf{x}$ in the design space as

$$m_i(\mathbf{x}) = [y(x_1, \ldots..,x_{i-1}, x_i + \Delta, x_{i+1}, \ldots\ldots, x_k) - y(\mathbf{x})] / \Delta,$$

where $\mathbf{x}$ is any value in Θ considered such that $\mathbf{x} + \Delta$ is still in Θ. The ultimate aim of the method is to estimate the mean and standard deviation of the finite distribution of $p^{k-1}[p - \Delta(p\text{-}1)]$ elementary effects associated with each input. Suppose, for the ith input factor this distribution F_i can be obtained by sampling $\mathbf{x}$ from Θ. The distribution F_i provides useful information about the impact of the ith input factor on the output through its mean μ and standard deviation σ. A large (absolute) measure of central tendency for F_i implies that an input has an important "overall" effect on the output. A high standard deviation indicates that the factor's effect is not constant, which may be explained by a factor interacting with other factors or a factor whose effect is non-linear. Morris' method considers the estimates of the means and standard deviations of these distributions as indicators to guide the selection of important input factors from the simulation.

In its simplest form, randomly selecting a value from F_i requires random selection of a value of each x_j (j = 1, 2, 3, ….., k) and evaluation of y twice, once at the selected values and again after increasing x_i by the quantity Δ; these two configurations would then yield one elementary effect. This could be repeated r times to produce a random sample of r elementary effects from F_i. However, if the procedure is performed for each input, it would be a random sample of r values from each F_i at a total cost of $n = 2rk$ runs (configurations). Morris defines the *economy* of a design as the number of elementary effects estimated by the design divided by the total number of required runs. So the economy of the sampling plan is 1/2, since a total of rk elementary effects are observed from the experiment. While the simplest form of Morris' design has an economy of 1/2, he suggests that a more economical design can be developed by using some runs in computing more than one elementary effect. This will be clear from the description of the method given in the following paragraphs.

To develop Morris' design, it is convenient to assume that p is even and $\Delta = p/[2(p - 1)]$. Overall, the design is based on the development of a matrix $\mathbf{B}^*$ whose rows represent the input vectors $\mathbf{x}$, and the corresponding experiment produces k elementary effects (one for each input factor) from $k + 1$ runs. This increases the economy of the design to $k/(k + 1)$. Morris (1991) showed that, with the assumption mentioned above, each of the $p^{k-1}[p - \Delta(p\text{-}1)] = p^k/2$ elementary effects for the ith input factor has an equal probability of being selected. The basic principle of Morris' design can be given as follows:

1. A "base" value $\mathbf{x}^*$ is randomly chosen for the vector $\mathbf{x}$, i.e. each component x_i of $\mathbf{x}^*$ is sampled from the set $\{0, 1/(p - 1), 2/(p\text{-}1), \ldots, 1 - \Delta\}$. Note, however, that the model is not evaluated at $\mathbf{x}^*$.
2. At least one of the k components of $\mathbf{x}^*$ are increased by Δ, leading to a vector $\mathbf{x}^{(1)}$ that is still in Θ. This may constrain the choice of components to be increased.

3. The estimated elementary effect of the *i*th component of $\mathbf{x}^{(1)}$ is

$$m_i(\mathbf{x}^{(1)}) = [y(x_1^{(1)}, \ldots..,x^{(1)}_{i-1}, x^{(1)}_i + \Delta, x^{(1)}_{i+1}, \ldots\ldots, x^{(1)}_k) - y(\mathbf{x}^{(1)})] / \Delta$$

if $\mathbf{x}^{(1)}$ has been increased by Δ, or

$$m_i(\mathbf{x}^{(1)}) = [y(\mathbf{x}^{(1)}) - y(x_1^{(1)}, \ldots..,x^{(1)}_{i-1}, x^{(1)}_i - \Delta, x^{(1)}_{i+1}, \ldots\ldots, x^{(1)}_k)] / \Delta$$

if $\mathbf{x}^{(1)}$ has been decreased by Δ.

4. Let $\mathbf{x}^{(2)}$ be the new sampling point $(x_1^{(1)}, \ldots..,x^{(1)}_{i-1}, x^{(1)}_i \pm \Delta, x^{(1)}_{i+1}, \ldots\ldots, x^{(1)}_k)$ which differs from $\mathbf{x}^{(1)}$ in its *i*th component. Select a third sampling point $\mathbf{x}^{(3)}$ such that $\mathbf{x}^{(3)}$ differs from $\mathbf{x}^{(2)}$ for only one component *j* (for any $j \neq i$). Either $x_j^{(3)} = x_j^{(2)} + \Delta$ or $x_j^{(3)} = x_j^{(2)} - \Delta$. Then the estimated elementary effect for the factor *j* is

$$m_j(\mathbf{x}^{(2)}) = [y(\mathbf{x}^{(3)}) - y(\mathbf{x}^{(2)})] / \Delta$$

if $\Delta > 0$, or

$$m_j(\mathbf{x}^{(2)}) = [y(\mathbf{x}^{(2)}) - y(\mathbf{x}^{(3)})] / \Delta$$

otherwise.

The above step (4) is repeated until a succession of $k+1$ sampling points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(k+1)}$ is produced such that two consecutive points differ in just one component. That means, to estimate one elementary effect for each factor, any component *i* of the "base vector $\mathbf{x}^*$" has to be increased by Δ at least once. The successive vectors $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots.., \mathbf{x}^{(k+1)}$ define a *trajectory* in the parameter space, and a matrix $\mathbf{B}^*$.

The matrix $\mathbf{B}^*$ is usually referred to as the *orientation matrix*. Its rows are the above mentioned vectors $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots.., \mathbf{x}^{(k+1)}$. In fact, the matrix corresponds to a trajectory of *k* steps in the parameter space with starting point $\mathbf{x}^{(1)}$, and gives one elementary effect per factor. The construction of the matrix, $\mathbf{B}^*$ involves the following steps: first, a $(k+1) \times k$ matrix, **B,** has been selected with elements 0s and 1s such that for every column there are two rows of **B** that differ in only one element. **B** may be chosen conveniently to be a strictly lower triangular matrix of 1s. Consider the transposed matrix given by

$$B' = M_{k+1,1}\, x^* + \Delta B,$$

where $M_{k+1,1}$ is a $(k+1) \times k$ matrix of 1s, and x^* represents a randomly chosen "base value" of x. Since the matrix B' can provide *k* elementary effects (one per input factor) at the cost of $(k+1)$ runs, it could be used as a design matrix. But, the limitation with B' lies in the fact that these *k* elementary effects are not randomly selected. A randomised version of Morris' design is given by:

$$B^* = (M_{k+1,1}\, x^* + (\Delta/2)\,[(2B - M_{k+1,k})D^* + M_{k+1,k}])P^*,$$

where $\mathbf{D}^*$ is a *k*-dimensional diagonal matrix with elements either +1 or –1 with equal probability, and $\mathbf{P}^*$ is a $k \times k$ random permutation matrix, i.e. each row and each column contain exactly one element equal to 1, while all other elements equal 0. A numerical example showing the derivation of an orientation matrix using Morris' OAT design is given in the Appendix. In order to estimate the mean and variance of the distribution F_i ($i = 1, 2, \ldots., k$), Morris considered a random sample of *r* elements, i.e., *r* mutually independent orientation matrices (corresponding to *r* different trajectories, each with different base vector). A single orientation matrix gives one elementary effect per factor, so the *r* matrices together give *rk* elementary effects. As a result, the design matrix for the entire experiment becomes

$$\mathbf{X} = \begin{bmatrix} \mathbf{B}_1^* \\ \mathbf{B}_2^* \\ \ldots \\ \mathbf{B}_r^* \end{bmatrix}.$$

The design provides *k* correlated estimators per trajectory, whereas the *r* independent trajectories provide *r* independent estimators. So, for each of the *k* factors, the mean and standard deviation can be estimated using the classical estimators of an independent random sample, whose values are from *r* mutually independent orientation matrices.

The main advantage of Morris' design is its relatively low computational cost. For the entire design **X**, *r* elementary effects are produced for each input factor at a total cost of $r(k+1)$ runs, which is a linear function of the number of factors involved. So, the economy of the sampling plan increases to $rk / r(k+1) = k/(k+1)$. Although the design does not produce estimates for individual interactions among different factors, it can provide information on whether any significant interaction exists, rather than giving specific information about the identity of the interaction.

3 CASE STUDY: A SCENARIO FROM A STOCHASTIC COMBAT SIMULATION MODEL, SIMBAT

SIMBAT is an Operational Analysis (OA) tool used by the UK's Defence Science and Technology Laboratory (Dstl) (formerly the Centre for Defence Analysis, CDA) to allow simulation of Battlegroup (BG) and Brigade level engagements, in order to provide quick assessment of equipment performance and tactics. SIMBAT represents a closed, force-on-force, stochastic combat simulation that can reflect a conventional war-fighting environment involving two-sided encounters. This is achieved by simple representations of: The forces, The ground, Command and control

(C^2), Movement, Obstacles, Logistics support, Surveillance and target acquisition, Rules of engagement and direct fire, Indirect fire and Human factors.

More specifically, SIMBAT itself includes a number of models, such as models of movement, breakdowns, fatigue, fuel usage, direct fire and ammunition usage, indirect fire, and simple command decision-making. A particular scenario can be defined and configured by the users from a Graphical User Interface (GUI). This contains the characteristics of the respective scenario elements (e.g. weapon performance, movement characteristics etc.); parameters of the various models within SIMBAT (e.g. sighting data, update cycle periods, etc.); characteristics of the ground over which the scenario is run; force and command structures; and routes and objectives of the individual scenario elements. A single entity, called a 'unit', represents a group of armoured-vehicles or personnel and is the principal building block of a scenario. Every part of a unit is assumed to maintain proper communication with each other in order to achieve the same objectives together.

The scenario used for our experiment is a meeting engagement between two forces, denoted by Red and Blue, each of approximately battalion size. The development of the scenario was carried out using the guidelines mentioned in "User Guide" of the SIMBAT model v2.5.

The Blue side consisted of two tank squadrons, each equipped with 14 tanks, and two Armoured Infantry (AI) companies, one equipped with 14 IFVs (Infantry Fighting Vehicles) and the other with 10 IFVs. The Blue force also had a Close Reconnaissance Troop of 8 recce vehicles, a squadron of Armoured Engineers, a Close Air Defence (CAD) Troop equipped with High Velocity Missiles (HVM), and an anti-tank section.

The Red force consisted of two tank companies, each equipped with 10 tanks, one armoured infantry (AI) company, equipped with 10 IFVs, and a reconnaissance company consisting of various recce vehicles, each of which carried an anti-tank missile post. The characteristics of the tanks, IFVs and recce vehicles, etc. were different for the two forces.

The order of battle (ORBAT) for the scenario was based on these figures, although only those platforms that had a direct fire capability were actually included. This means that any values produced during the analysis would be based upon combat power only. For each of the armoured infantry companies, dismounted infantry were included and added to the scenario as an 'Embussed' unit.
The model was run in Batch Mode by associating an action on files with the *.smb file type in Windows Explorer. The Batch Mode then runs all the replications of the scenario specified in the associated Replication Definition File. In our experiment, we consider 40 replications using 40 different random number streams. A screen shot of the scenario is shown at Figure 1.

Within SIMBAT, a unit is the main element of a particular scenario, since units move, observe, fire and make decisions. A unit is composed of a number of components, which are the lowest levels represented in SIMBAT, and the direct fire weapons platforms. A component usually represents a vehicle or a grouping of infantry. When a unit has no components left, it is considered 'dead' and therefore disappears from the scenario.

4 FINDING IMPORTANT FACTORS FROM SIMBAT USING MORRIS' OAT DESIGN

The scenario involves a large number of both direct/indirect fire parameters and human factors. From these, we have considered 13 for the analysis, including 2 human factors— "Probability of Shock" and "Probability of Surprise". These factors are as follows:

1. Number of components in Blue force HQ
2. Number of tanks in each Blue tank squadron
3. Number of IFVs in Blue C Company
4. Number of IFVs in Blue D Company
5. Number of components in Blue reconnaissance troop
6. Number of components in Blue anti-tank section
7. Number of components in Red force HQ

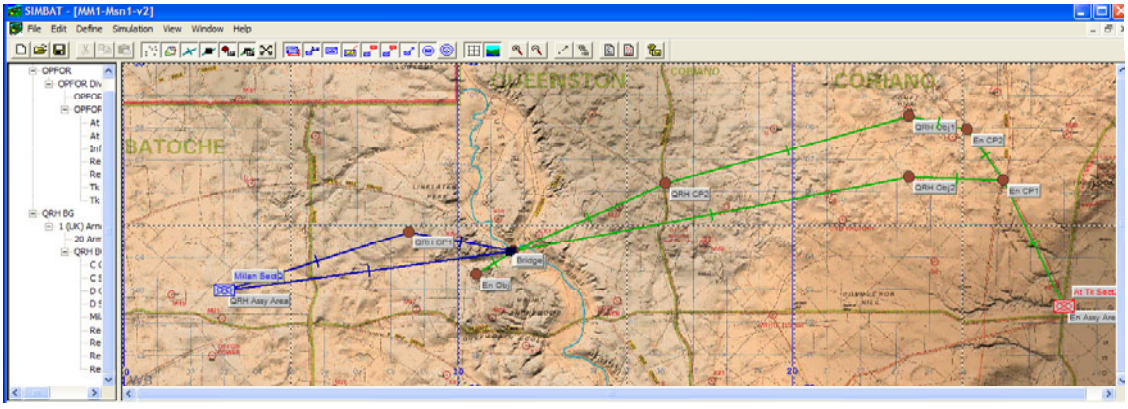

Figure 1: A Movement Network in the Scenario

8. Number of tanks in each Red tank company
9. Number of IFVs in Red Infantry company
10. Number of components in Red reconnaissance
11. Probability of Shock
12. Probability of Surprise, and
13. Unit Participation (in %).

Obviously the selection of factors depends on the problem we are looking at, and is also motivated by the use to which the metamodel will eventually be put. Whatever that use, it is likely that many factors can be kept at levels which will remain fixed, e.g. weapon characteristics. The simulation response considered here is Loss Exchange Ratio (LER), defined as:

LER=No. of Red components lost/ No. of Blue components lost, or
="Red" casualties / "Blue" casualties.

"Probability of Shock" represents the probability that in any close combat engagement or mini battle, units of a force will be shocked, thus reducing their SSKP. This paralysis within a force is introduced by the fear of an attack that can not be countered. For example, against infantry this could include an attack from air power or tanks. It takes a value between 0 and 1, and would apply to the force as a whole. Similarly, the "Probability of Surprise" is the probability that a force was surprised at the start of a particular scenario, thus reducing the SSKP in all mini battles and close combat engagements according to local force ratios. It is worth mentioning that the factor "Unit Participation" is a group factor, which has six sub-groups- small arm, canon, tank, machine gun, anti-tank and helicopter. "Unit Participation" represents the percentage of components in a unit, which are able to participate in direct fire using these six types of weapons. Measured in percentage (%), the factor can take values between 0 and 100 for each of its six sub-factors.

We have considered 6 different levels ($p = 6$) for each factor. So, the p values in the set {0, 1/(p-1), 2/(p-1), …., 1} would be equivalent to {0, 1/5, 2/5, 3/5, 4/5, 1} in our experiment. 13 factors and their corresponding levels are given in the following Table 1.

Then, we have applied Morris' factor-screening design, discussed in the preceding section, in order to see which factors have a significant influence on the simulation response. The parameters of the experiment are set to $p = 6$, $\Delta = p/[2(p-1)] = 3/5$ and $r = 6$. Six orientation matrices are generated according to the above design, and the respective elementary effects for 13 different factors per orientation matrix are estimated from the simulation response. Following Morris' (OAT) design, 14 simulation configurations are generated for each of the six orientation matrices. In each orientation matrix, the first row represents the "base" case (configuration) and the remaining 13 are used to determine the elementary effects for all 13 factors involved. Table 2 shows the elementary effects of 13 factors for 6 different trajectories, and the corresponding mean and variance of the distribution of F_i ($i = 1, 2, \ldots, k$).

The following figure shows that factors 2, 8, 10 and 13 are well separated from the other factors because of their higher values for mean and variance. Factors 5, 6, 11 and12 also show higher values for mean and variance in comparison with factors 1, 3, 4, 7 and 9. So, we can con-

Table 1: Levels of 13 Different Factors Used to Develop Morris' OAT Design

Factors	Levels					
	0	1/5	2/5	3/5	4/5	1
1. Number of components in Blue force HQ	1	2	3	4	5	6
2. Number of tanks in each Blue tank squadron	6	8	10	12	14	16
3. Number of IFVs in Blue C company	6	9	11	13	16	18
4. Number of IFVs in Blue D company	4	6	8	10	12	14
5. Number of components in Blue reconnaissance troop	4	8	12	16	20	24
6. Number of components in Blue anti-tank section	1	2	3	4	5	6
7. Number of components in Red force HQ	1	2	3	4	5	6
8. Number of tanks in each Red tank company	6	9	11	3	15	18
9. Number of IFVs in Red Infantry company	5	7	9	11	13	15
10. Number of components in Red reconnaissance company	6	10	14	18	22	26
11. Probability of Shock	0	0.2	0.4	0.6	0.8	1
12. Probability of Surprise	0	0.2	0.4	0.6	0.8	1
13. Unit Participation (in %)						
Small Arm	2	6	8	10	12	14
Cannon	8	11	14	17	20	23
Tank	8	11	14	17	20	23
Machine Gun	8	11	14	17	20	23
Anti-tank	12	19	26	33	40	47
Helicopter	12	19	26	33	40	47

Table 2: Elementary Effects of Various Factors for Six Different Trajectories

Factors	r =1	r =2	r = 3	r = 4	r = 5	r = 6	Mean	Variance
1	0.07	0.03	0.07	0.1	0.15	0.07	0.08	0.002
2	0.58	0.5	0.62	0.22	0.33	0.45	0.45	0.023
3	0.07	0.08	0.1	0.02	0.05	0.13	0.08	0.001
4	0.05	0.08	0	0.05	0.12	0.03	0.06	0.002
5	0.05	0.2	0.05	0.1	0.08	0.180	0.11	0.004
6	0.17	0.28	0.05	0.12	0.17	0.080	0.15	0.007
7	0.03	0.02	0.03	0.08	0.02	0.020	0.03	0.001
8	0.28	0.23	0.1	0.3	0.47	0.050	0.24	0.023
9	0.07	0.12	0.08	0.08	0.02	0.020	0.07	0.002
10	0.45	0.18	0.53	0.42	0.43	0.370	0.40	0.014
11	0.15	0.12	0.12	0.05	0.33	0.180	0.16	0.009
12	0.1	0.17	0.17	0.03	0.2	0.050	0.12	0.005
13	0.03	0.32	0.05	0.37	0.2	0.170	0.19	0.019

clude that factors 1, 3, 4, 7 and 9 are relatively unimportant in this scenario. The remaining 8 factors are important and of these factors, 2, 8, 10 and 13 have the most significant effects on the simulation response.

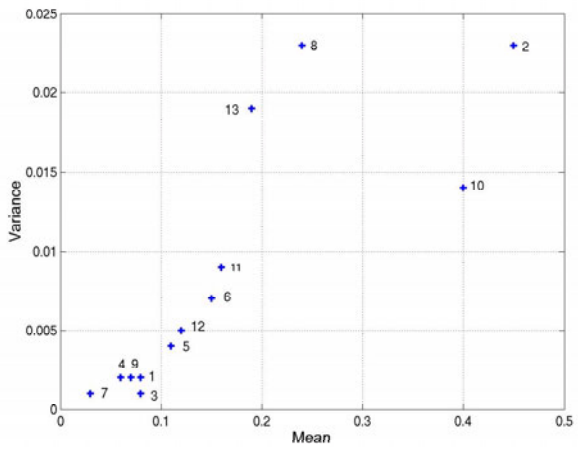

Figure 2: Estimated Mean and Variance of the Distribution of Elementary Effects for 13 Factors

5 DEVELOPING SIMULATION METAMODELS

The most important factors to emerge from the screening phase are then used for metamodel development. These factors are as follows: Number of tanks in each Blue tank squadron, Number of components in Blue reconnaissance troop, Number of components in Blue anti-tank section, Number of tanks in each Red tank company, Number of components in Red reconnaissance company, Probability of shock, Probability of surprise and Unit participation. Since the response surface of the underlying simulation is complex and not well behaved, a non-linear function approximation artificial neural network (ANN) is considered as an appropriate metamodeling method. A modified-Latin Hypercube Design (LHD) is employed to generate the required simulation configurations to develop candidate metamodels. The authors have found this design to perform better than other standard factorial or random sampling designs while developing metamodels from two other underlying simulations, as described in a previous article (Alam, McNaught and Ringrose, 2004a). Using this design, a total of 200 configurations were generated from within the design space defined by the 8 important factors. These configurations provide a training data set for developing the neural networks. Two further random data sets, each of size 100, were generated from within the design space, and employed as validation and test sets. For all training, validation and test data sets, each configuration was replicated 40 times with 40 different random number streams to output the simulation response. The average of the 40 LER values is considered as the final simulation output for that configuration. A neural network is then constructed which is trained on the simulation response, LER. Three approaches for neural network metamodels are considered:

1. The training set is made up of 200 simulation configurations, based on a modified-LHD, and the network is trained on the simulation response, LER.
2. The training set is made up of 200 simulation configurations, based on a modified-LHD. A separate network has been employed, and trained on each of the two force casualties, instead of the LER values. The predictions from the two force casualties are then combined to calculate the predicted LER.
3. The training set is made up of 200 simulation configurations, based on a modified-LHD. Instead of two separate networks, a single network is employed, and trained on each of the two force casualties. The simultaneous predictions of the network for the two force casualties are combined to give the LER.

Each approach uses the same validation and test sets, each consisting of 100 uniformly randomly sampled simulation configurations generated within the design space.

5.1 Separate Neural Network Metamodels for Individual Force Casualties

Neural network metamodels are also developed to predict individual force casualties. A separate network has been constructed for each of the two force casualties, Blue and Red. The neural network predictions for these individual force casualties are then combined to predict the LER. We have already observed that this indirect way of estimating LER might be of interest, and helpful in understanding complex combat simulation models (Alam, 2003). Finally, the predictions of force casualties from the two networks are combined to predict the LER.

5.2 A Single Neural Network for Simultaneous Prediction of the Two Force Casualties

A further experiment is carried out by constructing a single network, rather than two separate networks, to predict both force casualties mentioned in the preceding paragraph. Network predictions are then combined to estimate the LER. Neural networks with simultaneous predictions of targets, in this case Red and Blue force casualties, are computationally less expensive, but have the disadvantage that multiple output variables can interfere with each other's learning. The basic difference between a single network for simultaneous prediction of two targets and a separate network with only one of these two targets is that the first one involves more unknown parameters (bias and weights) than the second one. The reason for employing such a single network is to save time developing multiple networks.

A three-layer, feed-forward architecture (or multilayer perceptron) is adopted to construct the required networks for the above three approaches. For each approach, the networks have the same number (eight) of input nodes, representing the eight input factors, but they have a different number of output nodes, depending on the number of estimates being made by the networks. Since in approach 1, the network has one target factor, LER, it contains one output node. In approaches 2 and 3, however, the network has two target factors, the Blue and Red force casualties, so two output nodes are required. We have trained the networks using different numbers of hidden nodes, which is generally based on trial-and-error. However, the best performing network observed across the various approaches employed in this study has ten hidden nodes. Using a back-propagation training algorithm, all of the networks are trained to a normalized error tolerance of 1% and to a maximum of 200 epochs (passes through the training set). For each of the networks, several trials are conducted to find the architecture that minimizes the MSE between the target values and the network-predicted values. The performance of a particular network is evaluated with the associated test configurations and observed through three predictive measures, MSEP (Mean Squared Error of Prediction), PDRPE (Percentage Distribution of Relative Prediction Error) and MAPD (Mean Absolute Percentage Deviation). Relative Prediction Error (RPE) is defined as:

$$RPE = \frac{\hat{Y}_r}{Y_r},$$

where Y_r is the known target value (simulation response) from the independent test data set, and $\hat{Y}_r$ is the corresponding metamodel output or prediction. Generally, of course, we reject metamodels for which the RPEs are too far from 1, but this is a subjective decision and context dependent. Relative prediction errors (RPE) for each metamodel were derived in conjunction with the test data set. In Table 3, Percentage Distribution of Relative Prediction Error (PDRPE) shows the percentage of test data points for which the RPE falls between 0.8 and 1.2 (predicted value within 20% of the true value) and between 0.9 and 1.1 (predicted value within 10% of the true value). Another measure of performance is the Mean Squared Error of Prediction (MSEP), defined as:

$$MSEP = \frac{1}{N}\sum_{r=1}^{N}(Y_r - \hat{Y}_r)^2, \text{ (where } N = 100).$$

This is evaluated from the independent test cases for each of the developed metamodels. The prime advantage of the MSEP lies in its ability to incorporate a measure of both the variance and square of the bias of the prediction errors. The Mean Absolute Percentage Deviation (MAPD), which is defined as:

$$MAPD = \frac{1}{N}\sum_{r=1}^{N}|[\hat{Y}_r - Y_r]/Y_r|,$$

is also considered as it provides an easy-to-understand performance measure which can also be used for comparative reference between different studies. Depending on the degree of precision desired for a particular problem, a reasonable value for each of these measures may vary.

More detailed descriptions of the development of ANN metamodels are provided in two previous articles (Alam, McNaught and Ringrose 2004a; 2004b), so are not repeated here. All neural networks mentioned in this article are constructed using the MATLAB neural networks toolbox.

Table 3 presents the performance of ANN metamodels that are developed from the eight important factors, identi-

Table 3: Evaluation of Neural Network Metamodels Developed to Predict LER from Three Different Approaches

NN Metamodels	MSEP	PDRPE (within 20 %)	PDRPE (within 10 %)	MAPD
Approach 1	0.0052	95	75	7.98
Approach 2	0.0072	93	69	8.85
Approach 3	0.0090	92	63	10.17

fied by screening, in the previous section. To predict the simulation response, LER, neural network metamodels are developed using three different approaches, employing both direct and indirect ways of deriving the LER values.

For the particular scenario considered in this study, the networks developed from all three approaches show quite encouraging results. However, the best performance is observed for the one which is trained on the LER values directly (approach 1). Both the MSEP and the MAPD values, 0.0052 and 7.98 %, respectively, are lowest for this network, while the PDRPE values are highest. Approach 3, where a single neural network is developed to predict the two force casualty values simultaneously, before combining these to calculate the LER, displays the poorest performance of the three.

6 CONCLUSIONS

In this paper, we have demonstrated the use of Morris' randomized OAT design as a potential factor screening method to be used in developing simulation metamodels. In the literature, there are several methods for screening in simulation. However, each method is subject to some constraint in one aspect or another. The use of any particular method depends on the validity of its assumptions and on the behaviour of the system under study. Tradeoffs may be required by the experimenter when choosing a screening method. From the example problem considered, this study demonstrates that Morris, OAT design can be applied to a stochastic combat simulation. For this application, the assumptions required by more conventional factor screening methods, such as SB, are not satisfied. The important factors derived from the screening are then used to develop ANN metamodels for approximating the complex input-output relationship of the simulation. When evaluated with an independent fresh data set from the design space, the derived metamodels were able to produce good predictive capabilities for estimating the simulation response.

APPENDIX

To develop an orientation matrix using Morris' design, suppose $p = 4$, $k = 3$ and $\Delta = 2/3$. That is we want to investigate three factors that may contain values in the set {0, 1/3, 2/3 and 1}. Suppose, a "base" value

$$\mathbf{x}^* = (1/3, 0, 1/3)$$

is randomly chosen such that each component x_i is being sampled from the set $\{0, 1/(p\text{-}1), \ldots\ldots, 1\text{-}\Delta\}$. Now, at the next step we increase one or more components of $\mathbf{x}^*$ by Δ such that the new vector $\mathbf{x}^{(1)}$ is still in Θ. If we increase just the first component of $\mathbf{x}^*$ (i.e. first factor) by Δ, then $\mathbf{x}^{(1)}$ will become as

$$\mathbf{x}^{(1)} = (1, 0, 1/3),$$

and suppose this factor combination provides the simulation response $y^{(1)}$. Now, in the following step, suppose we want to have an elementary effect for the third factor. So, we will increase the third component of $\mathbf{x}^{(1)}$ by Δ while the other two components are kept fixed. Thus the vector for this new factor combination is

$$\mathbf{x}^{(2)} = (1, 0, 1),$$

which gives the simulation response $y^{(2)}$. Then the elementary effect of the third factor is calculated as

$$E_3 = [y^{(2)} - y^{(1)}]/ \Delta.$$

Similarly, in the next step if we want to determine the elementary effect of the first factor, we need to have the factor-combination as

$$\mathbf{x}^{(3)} = (1/3, 0, 1),$$

to provide the simulation response $y^{(3)}$. So, the elementary effect of the first factor will be

$$E_1 = [y^{(2)} - y^{(3)}]/ \Delta.$$

Finally, for the second factor the vector of factor-combination will be

$$\mathbf{x}^{(4)} = (1/3, 2/3, 1),$$

which will produce the simulation response $y^{(4)}$, and the elementary effect of this factor is derived as

$$E_2 = [y^{(3)} - y^{(4)}]/ \Delta.$$

Ultimately, we have a randomized version of orientation matrix **B***, given as follows

$$\mathbf{B}^* = \begin{bmatrix} 1 & 0 & 1/3 \\ 1 & 0 & 1 \\ 1/3 & 0 & 1 \\ 1/3 & 2/3 & 1 \end{bmatrix}.$$

REFERENCES

Alam, F. M., K. R. McNaught, and T. J. Ringrose. 2004a. A comparison of experimental designs in the development of a neural network simulation metamodel. *Simulation Modelling: Practice and Theory*. (To appear).

Alam, F. M., K. R. McNaught, and T. J. Ringrose. 2004b. Developing Simulation Metamodels for a Maintenance Float System. In *Proceedings of the OR Society 2nd Simulation Workshop*, ed. S. C. Brailsford, L. Oakshott, S. Robinson and S. J. E. Taylor. 293-302. Birmingham, U.K. March 23-24.

Alam, F. M. 2003. Investigating the development of metamodels for computer simulation. Unpublished PhD dissertation, Engineering Systems Department, Cranfield University, U.K.

Andres, T. H., and W. C. Hajas. 1993. Using iterated fractional factorial design to screen parameters in sensitivity analysis of a probabilistic risk assessment model. In *Proceedings of the Joint International Conference on Mathematical methods and Supercomputing in Nuclear Applications*. ed. H. Küsters, E. Stein and W. Werner. 2: 328-337.

Barton, R. R.1992. Metamodels for simulation input-output relations. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain and J.R. Wilson. 289-299.

Bettonvil, B., and J. P. C. Kleijnen. 1997. Searching for important factors in simulation models with many factors: sequential bifurcation. *European Journal of Operational Research* 96: 180-194.

Campolongo, F, J. Kleijnen, and T. Andres. 2000. Screening methods in sensitivity analysis. In *Mathematical and statistical methods for sensitivity analysis of model output*. ed. A. Saltelli and K. Chan. Chichester, England: Wiley.

Clarke, S. M., J. H. Griebsch, and T. W. Simpson. 2003. Analysis of Support Vector Regression for approximation of complex engineering analyses. In *Proceedings of ASME 2003 Design Engineering Technical Conferences*, Chicago, Illinois USA, September 2-6.

Hurrion, R. D., and S. Birgil. 1999. A comparison of factorial and random experimental design methods for the development of regression and neural network simulation metamodels. *Journal of the Operational Research Society* 50: 1018-1033.

Kleijnen, J. P. C. 1987. *Statistical Tools for Simulation Practitioners*, New York: Marcel-Dekker.

Kleijnen, J. P. C., and R. G. Sargent. 2000. A methodology for fitting and validating metamodels in Simulation. *European Journal of Operational Research* 120: 14-29.

Morris, M. D. 1987. Two stage factor screening procedures using multiple grouping assignments. *Commun. Statist. Theory Meth*.16: 3051-3067.

Morris, M. D. 1991. Factorial sampling plans for preliminary computational experiments. *Technometrics* 33: 161-174.

AUTHOR BIOGRAPHIES

FASIHUL M. ALAM is a Research Fellow in the Department of Information Systems and Computing at the Brunel University, U.K. He has a BSc (Hons.) and an MSc in Statistics from the University of Chittagong, Bangladesh and a PhD in Operational Research from Cranfield University. His current research interests include simulation metamodeling, experimental design issues of simulation models, multivariate analysis and artificial neural network.

KEN R. McNAUGHT has a BSc (Hons.) in Physics and Astronomy from Glasgow University, an MSc in Operational Research from Strathclyde University and a PhD in Operational Research from Cranfield University. He is currently a lecturer in the Applied Mathematics and Operational Research Group at Cranfield University's Shrivenham campus. His research interests include combat modelling, simulation, decision analysis and intelligent decision support systems.

TREVOR J. RINGROSE is a Lecturer in the Applied Mathematics & Operational Research Group, Engineering Systems Department at the Cranfield University, U.K. He received a BSc (Hons.) in Computer Science and Statistics from Sheffield University in 1985 and a PhD in Statistics from the same university in 1990. His current research interests include multivariate, resampling and response surface methods.