ITERATIVE TRAINING TO IMPROVE NEURAL NETWORK METAMODELS OF SIMULATED SYSTEMS

SANIYE BURCU OZSERIM AND ALICE E. SMITH
Department of Industrial Engineering
University of Pittsburgh
Pittsburgh, PA 15261
aesmith@engrng.pitt.edu

LT. COL. ROBERT A. KILMER
Center for Strategic Leadership
Army War College
Carlisle, PA 17013-5049
kilmerr@csl-emh1.army.mil

ABSTRACT:
This paper explores two alternatives to the traditional supervised training method of initially selecting the complete training sample randomly from the range of possible values. Both alternative methods have the same central idea of using more patterns where the input/output relationship is complex. One method estimates complexity by curvature of the response surface, while the other, "iterative training," estimates complexity by training errors. This latter approach requires virtually no additional computational effort and is shown to generally result in more precise models. The approaches are illustrated on constructing a neural network metamodel of a discrete event stochastic simulation of an inventory system.

Kilmer [3-6] used artificial neural networks to construct empirical metamodels of computer simulations. The problem that he successfully addressed was how to perform basic simulation tasks, such as prediction and comparison of alternatives, with neural approximations of a computer simulation. The main theme of this paper is to investigate two alternative approaches to selecting training patterns for the same purpose, i.e., to approximate computer simulations. This problem was selected because obtaining the training and testing pairs for a stochastic computer simulation is computationally costly, thus motivating the need to form the most accurate neural network model from a limited data set. In both new training approaches, learning is iterative. A subset of the training data is used to develop an initial neural network. The complexity of the relationship the network needs to learn is estimated by either looking at the response surface on a regional basis or by identifying regions with large training errors. The assumption is that a more complex region should be allocated more training examples, and the succeeding iterations add training patterns disproportionately from the complex region(s).

This general theme has been the subject of previous research. Ludik and

Cloete [8], explored the effect of various training strategies for backpropagation training using simple recurrent and temporal autoassociation neural networks. They demonstrated that training on increasingly more complex combined subsets reduces the number of weight updates to reach a given error criterion, suggesting that easier subtasks should be learned first. A similar approach was used by Sun et al. [10] and they also observed faster convergence and better error performance. Wann et al. [11] proposed a method to partition input patterns into groups based on distance from the class border. By using combinations of these groups, they were able to construct a variety of training sets including typical and border sets (i.e., those near class boundaries). Cheung, Lustig and Kornhauser [2] analyzed the backpropagation learning process in three stages split according to the behavior of the errors produced. These works suggest that examination of the training errors on certain observations can be used to improve the supervised neural network learning process beyond that achievable by simple batch training.

RESEARCH APPROACH AND RESULTS

The inventory simulation problem used was taken from Law and Kelton [7]. Kilmer [3-5] also studied this problem and it is his results using uniformly generated training patterns to which we compare. Note that each simulation "run" or "replication" results in one value of the response variable, total cost, and this response depends on both the values of the input variables, reorder point (*s*) and reorder quantity (*d*), and on the particular values taken on by the many random variables inherent in the simulation. At the beginning of each month, the inventory level is reviewed and a decision is made on how many items to order from the supplier. Figure 1 shows the expected value (mean of 10 replications) of the simulation over 420 combinations of *s* and *d*. This figure is useful when comparing the training patterns selected by the two alternative methods presented this paper. Note that the response surface reaches a minimum in the lower left quadrant, and this quadrant also contains the greatest changes in the response. The following are the parameters of the system as described in Kilmer [3].

$$D\ \text{(Demand)} = \begin{cases} 1 & with\quad probability\quad 1/6 \\ 2 & with\quad probability\quad 1/3 \\ 3 & with\quad probability\quad 1/3 \\ 4 & with\quad probability\quad 1/6 \end{cases}$$

t_d (time between demands)~exp($1/\lambda$=0.1 month)
I = inventory level (I_o = 60)
s = reorder point
d = reorder quantity when $I = s$

t_o = time between order decisions = 1 month
t_L = time lag for delivery (U [0.5 to 1] month)
k = setup cost = \$32.00
i = incremental cost per item ordered = \$3.00
h = holding cost = \$1.00 / month
u = underage cost = \$5.00 / month
n = time horizon = 120 months

We compare a total of eight networks. The first four are trained on the expected output (response) over ten simulation runs while the next four are trained on each of the ten responses. Each network is trained from the same sized set of training patterns (25 input/output pairs) and the network architectures (2-2-2-1), learning parameters, termination criteria (to 0.10 or 5000 epochs) and initial weights are identical. These architecture and training parameter choices were identical to those in Kilmer [3-4], who established them after experimentation. We compare the final error over a separate test set of 17 observations, where these observations include both interpolations (11 points) and extrapolations, i.e., values beyond any of those in the training set, (6 points). The first network is trained as reported in Kilmer [3-4] with training patterns evenly distributed throughout the variable ranges, and training done in one iteration on the whole training set of 25. This is the traditional method of training, using a randomly chosen set in one batch. The second network uses an initial training set of 16 patterns then adds 9 patterns in a second iteration, chosen by response surface curvature analysis. The third network uses the iterative approach based on training error with a first iteration of 16 patterns and a second iteration of 9 patterns. The fourth network also uses the iterative approach with a first iteration of 16 patterns, a second iteration of 4 patterns and a third iteration of 5 patterns. The fifth through eighth networks are the same, but trained on individual replications rather than mean responses.

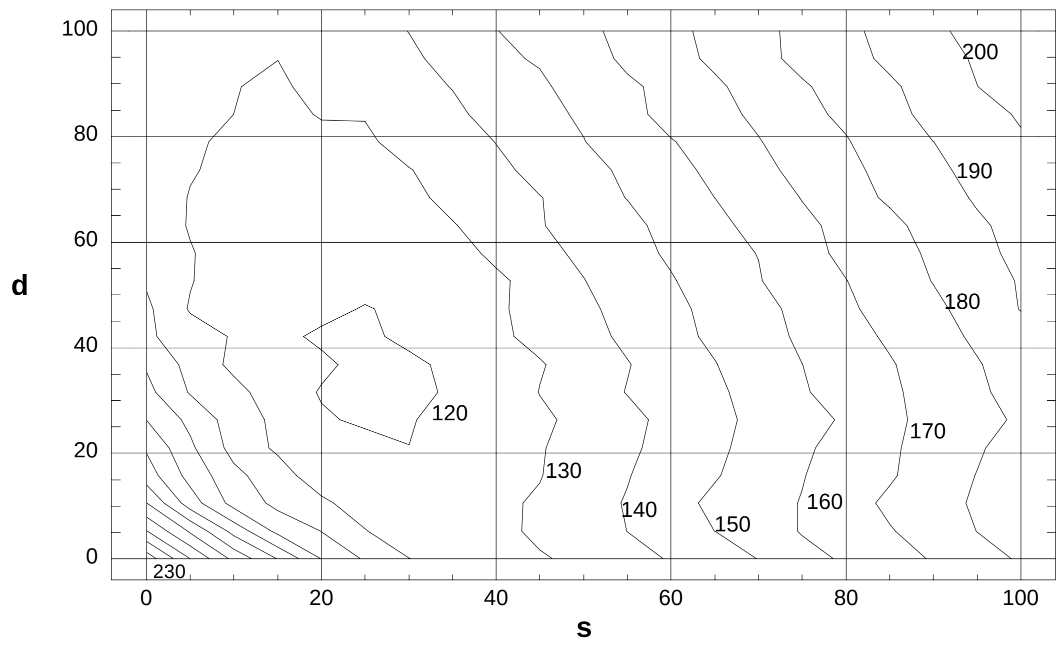

Figure 1: Simulation expected response (total cost) to *s* and *d* over 420 simulation runs.

Experiments on Using Expected Values as Targets

Network 1 - Uniformly Selected*:* Figure 2 shows the training (stars) and testing (squares) patterns for the network trained on 25 equally distant patterns. Note that these are the testing patterns for all networks studied in this paper.

Network 2 - Response Surface Method: A measure of regional curvature was used as an approximation to complexity. That is, a linear (planar) response surface is assumed to be simple, while curvature indicates that the region may be harder to learn. Regions are taken to be equally sized in each input dimension. We constructed four regions by dividing both *s* and *d* in half (2^2=4), i.e., four quadrants. Regions are coded by the two levels of *s* and *d*, e.g. region (1,1) is the lower left and region (2,1) is lower right. The second derivative of a simulation is impossible to evaluate since there is no explicit form. As a surrogate, response surface methodology was used to select 9 additional training patterns to add to the 16 existing uniformly distributed patterns. Curvature and interaction checks are made independently for each region using a 2^2 factorial with 3 center points (Figure 3).

The design chosen allows the planar model to be efficiently fitted, allows checks to be made to determine whether the planar model is representationally adequate, and provides some estimate of experimental error. The least squares estimate of β_0 is the mean of all 7 observations. $\beta_1 = 1/4 [(-1 * r_1) + (1 * r_2) + (-1 * r_3) + (1* r_4)]$. The values; r_1, r_2, r_3 and r_4, stand for the response of the first four runs for region (1,1) respectively. Similarly, the least squares estimate of β_2 is calculated in the same way. An estimate of the experimental error variance is obtained from the 3 replications at the center. It was very approximate because there are only 2 degrees of freedom. Standard errors of coefficients are calculated by using this estimate.

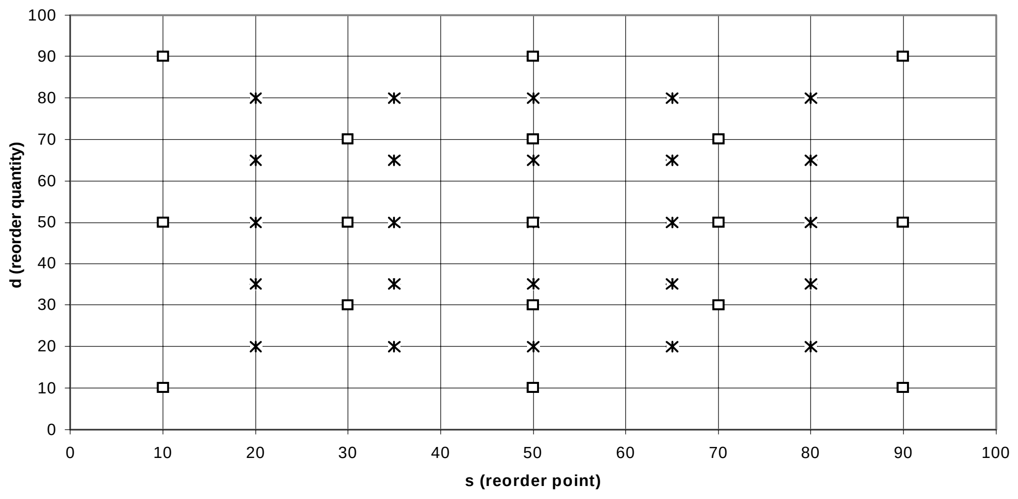

Figure 2: Training and testing patterns for network 1.

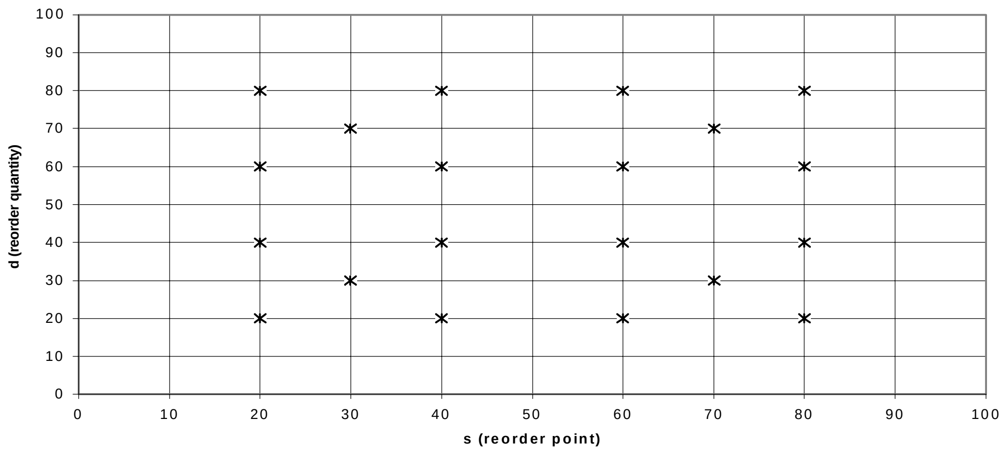

Figure 3: Points used for response surface methodology.

The effects of the variables are assumed to be additive. The coefficient β_{12} of an added cross product term x_1x_2 in the model is used to measure the interaction between the variables. This coefficient $= 1/4 [(1*r_1) + (-1*r_2) + (-1*r_3) + (1*r_4)]$. Another measure of planarity is provided by comparing the average of the 4 points of the 2^2 factorial with the average at the center of the design. In [1] it is shown that if β_{11} and β_{22} are coefficients of the terms x_1^2 and x_2^2, this curvature measure will be an estimate of $\beta_{11} + \beta_{22}$. The estimate of the overall curvature is $\beta_{11} + \beta_{22} = 1/4 (r_1 + r_2 + r_3 + r_4) - 1/3 (r_5 + r_6 + r_7)$. The values r_5, r_6 and r_7 stand for responses of the center values. In summary, the planarity checking functions are ($\beta_{12} \pm s_{\beta 12)}$ and ($\beta_{11} + \beta_{22} \pm s_{(\beta 11 + \beta 22)}$). If both of the functions include "0" then the surface is planar, otherwise, the surface is not planar.

Table 1 shows the curvature and interaction check results. Only region (1,1) (lower left) was non-planar. Therefore 8 of the 9 additional patterns were selected randomly from this region. Since the ranges for both the interaction and curvature checks for region (1,2) (upper left) were wider than those of regions (2,1) and (2,2), 1 additional pattern was selected randomly from region (1,2) (upper left).

TABLE 1: INTERACTION AND CURVATURE CHECK RESULTS

Region	Interaction check	Curvature check	Decision	# of points
(1,1)	(2.12, -3.45)	(10.70, 2.18)	non-planar	8
(1,2)	(1.36, -1.44)	(2.73, -1.55)	planar	1
(2,1)	(0.46, -0.94)	(1.29, -0.85)	planar	0
(2,2)	(1.29, -0.57)	(0.72, -2.13)	planar	0

<u>Network 3 - One Iteration</u>: First, an initial 16 equally distant training patterns (4 in each region) are used to trained the network. The region with the largest training mean absolute error (MAE = Total Absolute Error / number of training points) is selected to receive the next training pattern, randomly selected from that region. Then the MAE is reduced by adding 1 to the denominator (it is now 4+1 = 5). The second additional training point goes to the region with the highest MAE, and so on until 9 points are chosen to add to the 16 to form a total set of 25. The denominator to figure the region's MAE is incremented by 1 for each additional training pattern it receives. The network continues training on this combined set until the termination criterion is met. The results of the initial training are shown followed by the selection of the 9 additional points:

region code	total absolute error	MAE	# points / region
(1,1)	58.46	14.61	4
(1,2)	36.72	9.18	4
(2,1)	22.47	5.61	4
(2,2)	34.87	8.71	4

1	new point from region (1,1)	(20,22)	expected new MAE in region	11.69
2	new point from region (1,1)	(30,21)	expected new MAE in region	9.74
3	new point from region (1,1)	(31,27)	expected new MAE in region	8.35
4	new point from region (1,2)	(36,56)	expected new MAE in region	7.34
5	new point from region (2,2)	(71,78)	expected new MAE in region	6.97
6	new point from region (1,1)	(28,33)	expected new MAE in region	7.30
7	new point from region (1,2)	(23,71)	expected new MAE in region	6.12
8	new point from region (1,1)	(37,21)	expected new MAE in region	6.49
9	new point from region (2,2)	(55,74)	expected new MAE in region	5.81

<u>Network 4 - Two Iterations</u>: Network 4 was trained in two iterations. As for the third network, 16 initial equally distant patterns were used to train the initial network. Then 4 additional training patterns were added and the network continued training for the first iteration. For the second iteration, an additional 5 training patterns were selected and the network continued training on the total set of 25, as shown below.

ITERATION 1:

region code	total absolute error	MAE	# points / region
(1,1)	58.46	14.61	4
(1,2)	36.72	9.18	4
(2,1)	22.47	5.61	4
(2,2)	34.87	8.71	4

1	new point from region (1,1)	(20,22)	expected new MAE in region	11.69
2	new point from region (1,1)	(30,21)	expected new MAE in region	9.74
3	new point from region (1,1)	(31,27)	expected new MAE in region	8.35
4	new point from region (1,2)	(36,56)	expected new MAE in region	7.34

ITERATION 2:

region code	total absolute error	MAE	points / region
(1,1)	89.44	12.77	7
(1,2)	38.48	7.69	5
(2,1)	20.76	5.19	4
(2,2)	34.95	8.73	4

1	new point from region (1,1)	(22,22)	expected new MAE in region 11.18
2	new point from region (1,1)	(33,23)	expected new MAE in region 9.93
3	new point from region (1,1)	(36,28)	expected new MAE in region 8.94
4	new point from region (1,1)	(36,25)	expected new MAE in region 8.13
5	new point from region (2,2)	(71,79)	expected new MAE in region 6.90

The results of these experiments over the 17 test points are summarized in Table 2. The performance of the networks trained iteratively (networks 3 and 4) are better than the network trained on equally distant patterns (network 1). The percent improvement is the improvement in internal MAE over network 1, the conventional training strategy.

TABLE 2: TOTAL AND INTERNAL TESTING MAE

	Total Testing MAE	Internal Testing MAE	Percent Improvement
Network 1	12.10	9.39	-
Network 2	11.60	8.77	6.6%
Network 3	10.88	8.22	12.5%
Network 4	10.80	8.39	10.6%

Experiments on Using Simulation Replications as Targets

This section investigates networks trained on 10 replications of each pattern. Refer to Kilmer [3] for a complete discussion on expected value versus replication training.

Network 5 - Uniformly Selected: Network 5 was trained on 25 equally distant training patterns with 10 simulation replications made for each pattern. Therefore the total number of patterns to train the network was 250. The input training patterns (s,d) are same as the ones used to train network 1 in the preceding section.

Network 6 - Response Surface Method: Network 6 was trained on a total of 25 training patterns, which were also selected for network 2, but the training set included each individual replication response, rather than just the mean of the 10 responses.

Network 7 - One Iteration: For network 7, 160 total patterns (replications of 16 equally distant patterns) were used for initial training. After evaluating the region absolute errors, 9 additional patterns were selected. 10 replications of these additional 9 points were added to the existing 160 points and a total of 250 were used to train the final network, as shown below.

region code	total absolute error	MAE	# points / region
(1,1)	388.69	9.71	4×10 replications
(1,2)	382.88	9.57	4×10 replications
(2,1)	165.80	4.14	4×10 replications
(2,2)	273.57	6.83	4×10 replications

1	new point from region (1,1)	(23,23)	expected new MAE in region 7.77
2	new point from region (1,2)	(30,51)	expected new MAE in region 7.65
3	new point from region (1,1)	(31,27)	expected new MAE in region 6.47
4	new point from region (1,2)	(36,56)	expected new MAE in region 6.38
5	new point from region (2,2)	(71,79)	expected new MAE in region 5.47
6	new point from region (1,1)	(28,33)	expected new MAE in region 5.55
7	new point from region (1,2)	(23,71)	expected new MAE in region 5.46
8	new point from region (1,1)	(37,21)	expected new MAE in region 4.85
9	new point from region (2,2)	(55,74)	expected new MAE in region 4.55

Network 8 - Two Iterations: This network was like network 4, but used ten replications for each input pattern selected. The first iteration selected 4 additional points and the second iteration selected 5 more.

Iteration 1:

region code	total absolute error		MAE	# points / region
(1,1)	388.69		9.71	4 × 10 replications
(1,2)	382.88		9.57	4 × 10 replications
(2,1)	165.80		4.14	4 × 10 replications
(2,2)	273.57		6.83	4 × 10 replications

1	new point from region (1,1)	(23,23)	expected new MAE in region	7.77
2	new point from region (1,2)	(30,51)	expected new MAE in region	7.65
3	new point from region (1,1)	(31,27)	expected new MAE in region	6.47
4	new point from region (1,2)	(36,56)	expected new MAE in region	6.38

ITERATION 2:

region code	total absolute error		MAE	# points / region
(1,1)	539.54		8.99	6 × 10 replications
(1,2)	600.71		10.0	6 × 10 replications
(2,1)	160.67		4.01	4 × 10 replications
(2,2)	272.12		6.80	4 × 10 replications

1	new point from region (1,2)	(22,58)	expected new MAE in region	8.58
2	new point from region (1,1)	(33,21)	expected new MAE in region	7.70
3	new point from region (1,2)	(34,57)	expected new MAE in region	7.50
4	new point from region (1,1)	(38,24)	expected new MAE in region	6.74
5	new point from region (1,2)	(41,79)	expected new MAE in region	6.67

The results, as shown in Table 3, are similar to those from networks 1 through 4. The networks trained iteratively using training errors were superior, with two additional iterations the clear winner. The response surface technique did not fare well for this group of networks. Overall, the networks had lower error rates than the first four, confirming Kilmer's conclusion [3] that replication training is generally superior for small data sets.

TABLE 3: TESTING RESULTS OF NETWORKS TRAINED ON REPLICATIONS

	Total Testing MAE	Internal Testing MAE	Percent Improvement
Network 5	10.20	7.34	-
Network 6	10.28	7.53	-2.6%
Network 7	9.70	7.12	3.0%
Network 8	9.12	6.27	14.6%

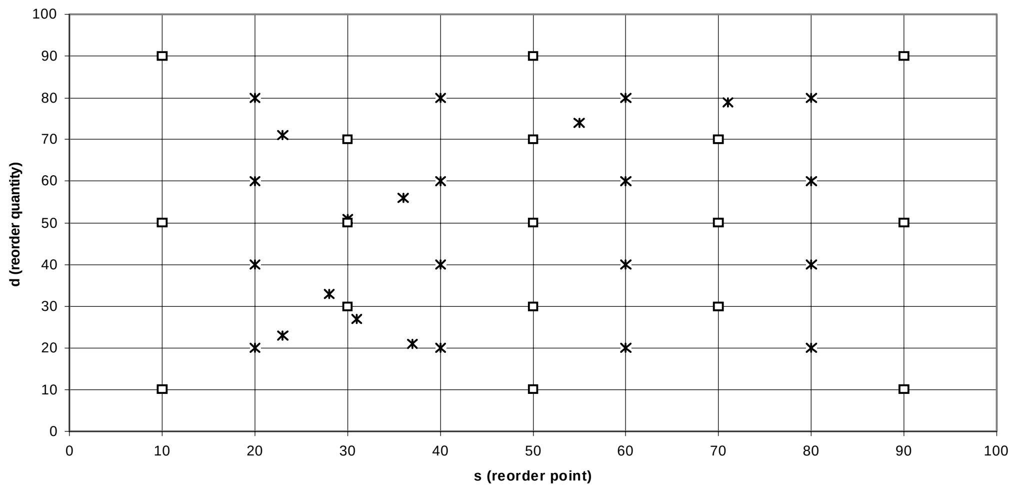

Figure 4: Typical training set selected by the iterative method.

DISCUSSION

In both of the experiments, the regions where *s* was between 20 and 40 and *d* was between 20 and 80 were the ones which indicated more complexity, *viz*. regions (1,1) and (1,2). Both of the alternative methodologies detected these regions as potential training bottlenecks, and therefore allocated them more training patterns in subsequent iterations. Figure 4 shows a typical training set selected by the iterative method. However, it is difficult and computationally expensive to approximate the simulation function by the response surface methodology. As the number of parameters grows, the complexity of approximating the second derivative increases. On the other hand, evaluation of training errors and iteratively training are easy and require hardly any additional computation. We have made further tests on the robustness of this method for larger problems [9] and have confirmed the general superiority of the iterative training method. The additional computational effort required, regardless of problem size, is minimal, which offers encouragement that this method can be applied to almost any neural network training problem with little *a priori* knowledge or experimentation.

ACKNOWLEDGEMENT

The second author gratefully acknowledges financial support from the National Science Foundation CAREER grant DMI-9502134.

REFERENCES

1. Box, G. E. P., W. G. Hunter and J. S. Hunter, *Statistics for Experimenters*, John Wiley & Sons, New York, 1978.
2. Cheung, R. A. M., I. Lustig and A. L. Kornhauser, "Relative effectiveness of training set patterns for back propagation," *Proceedings of the International Joint Conference on Neural Networks*, 1990, I-673-I-678.
3. Kilmer, R. A., *Artificial Neural Network Metamodels of Stochastic Computer Simulations*, unpublished Ph.D. dissertation, University of Pittsburgh, 1994.
4. Kilmer, R. A. and A. E. Smith, "Using artificial neural networks to approximate a discrete event stochastic simulation model," *Intelligent Engineering Systems Through Artificial Neural Networks, Volume 3*, (C. H. Dagli, L. I. Burke, B. R. Fernandez, J. Ghosh, editors), ASME Press, New York, 1993, 631-636.
5. Kilmer, R. A., A. E. Smith and L. J. Shuman, "Neural networks as a metamodeling technique for discrete event stochastic simulation," *Intelligent Engineering Systems Through Artificial Neural Networks, Volume 4*, (C. H. Dagli, B. R. Fernandez, J. Ghosh and R. T. S. Kumara, editors), ASME Press, New York, 1994, 1141-1146.
6. Kilmer, R. A., A. E. Smith and L. J. Shuman, "An emergency department simulation and a neural network metamodel," *Journal of the Society for Health Systems*, in press.
7. Law, A. M. and W. Kelton, *Simulation Modeling and Analysis Second Edition*, McGraw-Hill, New York, 1991.
8. Ludik, J. and I. Cloete, "Training schedules for improved convergence," *Proceedings of the 1993 International Joint Conference on Neural Networks*, 1993, 561-565.
9. Ozserim, S. B., *Improving Neural Network Simulation Metamodels Using Iterative Training*, unpublished M.S.I.E. Thesis, University of Pittsburgh, 1995.
10. Sun, X., B. Golden and E. Wasil, "The fine-tuned learning enhancement to the standard backpropagation algorithm," *Intelligent Engineering Systems Through Artificial Neural Networks Volume 5* (C. H. Dagli, M. Akay, C. L. P. Chen, B. R. Fernandez and J. Ghosh, editors), ASME Press, New York, 1995, 125-132.
11. Wann, M., T. Hediger and M. M. Greenbaum, "The influence of training sets on generalization in feed forward neural networks," *Proceedings of the 1993 International Joint Conference on Neural Networks*, 1993, 137-142.