An Algorithm to Determine Sample Sizes for Optimization with Artificial Neural Networks

Aroonsri Nuchitprasittichai and Selen Cremaschi
Dept. of Chemical Engineering, The University of Tulsa, 800 South Tucker Drive, Tulsa, OK 74104

DOI 10.1002/aic.13871
Published online July 3, 2012 in Wiley Online Library (wileyonlinelibrary.com).

This article presents an algorithm developed to determine the appropriate sample size for constructing accurate artificial neural networks as surrogate models in optimization problems. In the algorithm, two model evaluation methods—cross-validation and/or bootstrapping—are used to estimate the performance of various networks constructed with different sample sizes. The optimization of a CO_2 capture process with aqueous amines is used as the case study to illustrate the application of the algorithm. The output of the algorithm—the network constructed using the appropriate sample size—is used in a process synthesis optimization problem to test its accuracy. The results show that the model evaluation methods are successful in identifying the general trends of the underlying model and that objective function value of the optimum solution calculated using the surrogate model is within 1% of the actual value. © 2012 American Institute of Chemical Engineers *AIChE J,* 59: 805–812, 2013

Keywords: sample size determination, incremental Latin hypercube sampling, artificial neural networks, cross-validation, superstructure optimization

Introduction

Process synthesis is the determination of the optimal type and design of processing units along with the interconnection of these units to produce products given the raw materials.[1] Two major approaches can be used to solve the process synthesis problem. The first approach, the traditional sequential method, involves using heuristic rules. Another process synthesis approach is superstructure optimization, which is based on simultaneous optimization using mathematical programming. This leads to a design procedure in which a superstructure is developed that includes all of the possible pieces of equipment in the process and their interconnections. This strategy requires the use of discrete variables for the selection of equipment and some design specifications, along with the first-principle process models. The resulting mathematical programming model—a large-scale nonconvex mixed integer nonlinear program—is difficult to solve.[2] In an attempt to simplify the mathematical programming problem, several researchers have proposed frameworks that use surrogate models. Values of the input variables (design variables) used to create the model are populated using experimental designs, and the output values corresponding to these inputs are generated by process simulators. The surrogate models, which represent functional relationships between the input variables and the outputs, are then created by statistical combination of the input values and the corresponding simulated outputs.[3]

In general, the techniques for building surrogate models are response surface methodologies (RSMs), kriging interpolators, and artificial neural networks (ANNs). RSMs fit a local first- or second-order model regression surfaces to guide the optimization search by providing a descent direction for the objective function (assume minimization). In the area of chemical engineering, the literature presents the application of RSMs in optimization of supply chains,[4] industrial hydrogenation process,[5] solvent fermentation process,[6,7] CO_2 capture process,[8] and wet air oxidation process.[9] Because RSMs usually utilize first- or second-order functions, they lack the ability to describe local surfaces that cannot be adequately approximated with these first- or second-order functions. Another limitation of RSMs is that their application to high-dimensional problems is still unsuccessful.[4]

Kriging interpolators enable one to build accurate global approximations of a decision space.[10] They use statistical interpolations to estimate the adjustable parameters that reflect the importance of each variable. These adjustable parameters offer a good model representation.[11] Davis and Ierapetritou,[12] and Caballero and Grossmann[11] have developed and applied algorithms combining kriging and RSM to the optimization of black-box systems in chemical engineering. These algorithms are based on fitting a response surface using a kriging model as the surrogate model. The combined model increases the likelihood of finding the global optimum compared to RSM alone. Simpson et al.[10] have applied kriging and RSMs using the generalized reduced gradient algorithm to an aerospike nozzle design problem. They found that the kriging model yields smaller prediction errors in the optimum design than those obtained from the RSM. However, kriging models require more effort to generate when

Additional Supporting Information may be found in the online version of this article.

Correspondence concerning this article should be addressed to S. Cremaschi at selen-cremaschi@utulsa.edu

compared to simple response surface models, and the number of readily available software programs that can be used to fit the models in the field of engineering is still limited.

ANNs are successfully used currently as predictive tools and pattern recognition techniques.[13] ANNs are computational systems developed based on biological networks.[14] They are characterized by their large number of adjustable parameters (weights and biases). The main advantage of ANNs is that they learn from examples, so a problem solving algorithm is not required.[15] ANNs have been widely applied to many chemical engineering optimization problems, such as, superstructure optimization,[16,17] the optimization of Nylon-6,6 polymerization in a twin-screw extruder reactor,[18] the optimization of an acetic anhydride plant,[18] the optimization of gasoline and diesel production,[13] and the optimization of biodiesel production.[19]

The selection of a suitable sample size for constructing an accurate network is an important consideration for successful application of ANNs.[20,21] The number of sample points should be large enough for the ANN architecture to be able to adapt to the structure of the data. A larger number of sample points allows for a larger hidden layer, as well as greater nonlinearity. However, large sample sizes require large computational time during the ANNs training and data collection. Gueddar and Dua[22] proposed disaggregation–aggregation techniques for training the ANNs. In this technique, a full dataset was split into subsets. Each subset is used for training different sub-ANNs in each computer in a parallel computing platform. Then, a new ANN of the overall data is trained using the sub-ANNs. Their approach is successful in reduction of the computational time to obtain the ANNs. However, they did not consider the reduction of the computational time in data collection. In order to reduce the computational time in data collection and balance the computational time with the accuracy of the network, the appropriate sample size needs to be determined.

Devabhaktuni and Zhang[21] developed a neural network training-driven adaptive sampling algorithm focusing on data sampling techniques. The desired accuracy (average test error) of the neural network model is used as the stopping criterion. The algorithm starts with the consideration of the original bounded input region. The ANN is trained using the vertices of the region as the training set and is tested using the center point of the region as the test data. If the ANN model accuracy does not match the desired accuracy, the original region is then split into 2^n regions of equal volume, where n is the number of input variables. The training and test data are systematically generated, and the ANN is trained and tested for each region. If the neural network model accuracy does not match the desired accuracy, the region that has the highest test error is then spilt to 2^n regions again and is further sampled until the network model accuracy reaches the desired value. Although this approach improves the accuracy of the ANN in the entire bounded input region, for high number of input variables, splitting the region into 2^n subregions requires a large number of data points. Dhabak and Pandit[23] developed an adaptive sampling algorithm to generate an optimum sample size for training ANNs. The algorithm starts with an initial sample size, n_0, of 10% of the maximum sample size, n_{MAX}. The distribution of the next sample size is assumed to be Gaussian, and the next sample size is calculated by the formula $n_{i+1} = \mu + \sigma N$ where $3\sigma = (n_{MAX} - n_0)/2$, $\mu = n_i$, and N is a random number drawn from the distribution with zero mean and unity standard deviation. The algorithm terminates when the accuracy of the model is stabilized or the algorithm breaks when the sample size is higher than n_{MAX}. This algorithm requires the maximum sample size as one of the inputs and it is possible that the right n_{MAX} was not specified resulting in the algorithm's terminating without providing results. It is especially difficult to determine a priori maximum sample size for generating ANNs for superstructure optimization applications, because the appropriate sample size varies widely amongst unit operations, and with the inlet and outlet stream properties.

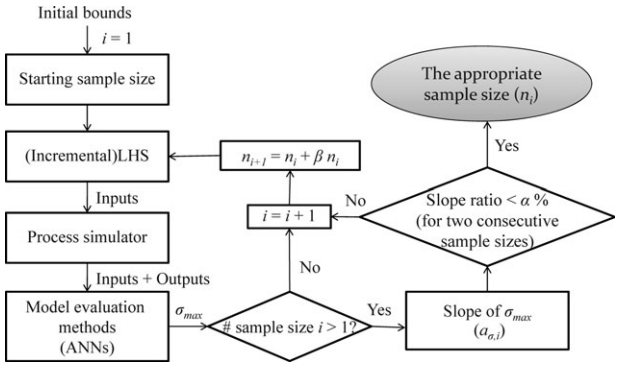

Figure 1. Sample size determination algorithm.

In this article, a new algorithm to determine an appropriate sample size for constructing accurate ANNs for superstructure optimization problems is presented. In the algorithm, model evaluation methods are employed as the tools to estimate the appropriate sample size. The algorithm utilizes Latin hypercube sampling (LHS) as the sampling technique. We introduce the optimization of a CO_2 capture process using aqueous amines with an objective to minimize the CO_2 capture costs as the case study. The case study illustrates the algorithm and compares the performance of two different model evaluation methods.

The details of the algorithm are introduced in "Methodology" Section. "Case Study: Optimization of CO_2 Capture Process with Aqueous Amines" Section applies the algorithm to the case study, which involves the optimization of the CO_2 capture process with aqueous amines. The results and discussion are provided in "Results and Discussion" Section. Finally, "Conclusions" Section provides conclusions.

Methodology

The overall methodology described in this section is used to construct an accurate ANN for superstructure optimization problems. An overview of the proposed sample size determination algorithm is shown in Figure 1. The inputs for the algorithm are: (1) the ranges of the decision variables, (2) the starting sample size, n_1, (3) proportion for sample size increments, β, and (4) a slope ratio percentage, α, which is used as the termination criterion.

The algorithm starts with the iteration counter (i) equal to one. Given the starting sample size, n_1, LHS is performed to generate a test matrix of the decision variables, that is, the input data. The process simulator is run according to the test matrix to obtain the output data. Input and output data are used to construct and estimate the performance of the ANN using model evaluation methods; cross-validation and bootstrapping are used as the model evaluation methods in this algorithm

("Model evaluation methods" Section). The maximum surrogate-model deviation, $\sigma_{\max}$, is calculated in this step.

If the iteration counter (i) is equal to one, it is increased and another sample size is generated by increasing the sample size by the proportion, β, of the current sample size. To minimize the additional necessary data collection at this stage, an incremental LHS (iLHS) ("iLHS" Section) approach is developed and used to generate the additional test matrix of decision variables. For $i > 1$, the slope of the maximum surrogate-model deviation, $a_{\sigma,i}$ corresponding to the current iteration counter, i, is calculated. Equation 1 shows the formula to calculate the slope of the maximum surrogate-model deviation, where n_i is the number of sample points corresponding to iteration i.

$$a_{\sigma,i} = \frac{|\sigma_{\max,i} - \sigma_{\max,i-1}|}{n_i - n_{i-1}} \tag{1}$$

The slope ratio percentage of the maximum surrogate-model deviation is utilized to determine whether the statistical quantity has stabilized or not. Because the surrogate-model deviation represents the accuracy of the surrogate model, that is, the ANN, the stabilization of the surrogate-model deviation means that further increases in the sample size do not improve the accuracy of the model. The slope ratio percentage of the maximum surrogate-model deviation is calculated by dividing the slope of iteration i ($a_{\sigma,i}$) by the highest slope obtained for all iterations ($a_{\sigma,\text{highest}}$) and then multiplying by 100. The lower the slope ratio percentage, the more likely it is that the value of the statistical quantity is stabilized. Therefore, a slope ratio percentage of an acceptable value, α, is used as the termination criterion to obtain the appropriate sample size for constructing the ANN.

If the slope ratio percentage is higher than α, the sample size is increased by a factor of β and the iteration counter is increased by one. With each sample size increment, the value of the maximum surrogate-model deviation changes until it stabilizes. The maximum surrogate-model deviation stabilizes when the slope ratio percentage is less than α for two consecutive sample sizes. The sample size at this point of stabilization is the appropriate sample size for constructing the ANN.

In what follows, a brief overview of the ANNs, two model evaluation techniques utilized in this study, and developed iLHS algorithm are discussed.

ANNs

In the algorithm, ANNs are used as powerful data modeling systems to represent the relationship between inputs and outputs. ANNs are able to learn directly from the data being modeled and able to represent the relationship of both linear and nonlinear models.[24]

A multilayer feed-forward network is commonly used in ANNs because of its mathematical simplicity. This property is especially important for superstructure optimization applications, where the purpose of utilizing ANNs is to reduce the mathematical complexity. The multilayer feed-forward network consists of three layers: an input layer, a middle layer, and an output layer. The middle layer consists of at least one hidden layer. Figure 2 shows a schematic of a multilayer feed-forward neural network with a single hidden layer. The data set for input and output layers are provided for the network. Each hidden layer contains a certain number of neurons, each of which has a transfer function that translates the input signals to output signals. In general, the available data are split into two sets. One set is used to train the neutral network, and the other set is used to test the performance of the trained network.[18]

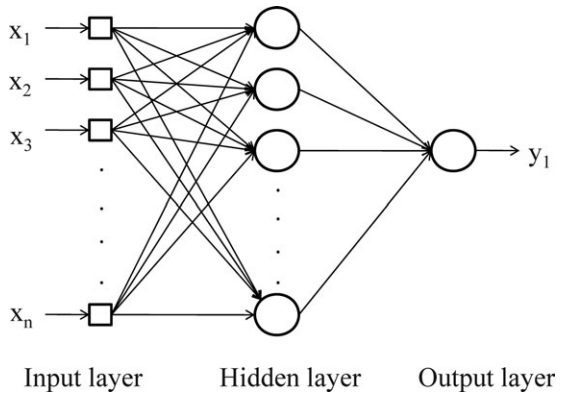

Figure 2. A schematic of a multilayer feed-forward neural network with a single hidden layer.

Back propagation (BP) is the most widely used technique in the training of multilayer feed-forward networks. A BP algorithm involves two phases, which are the forward and backward phases. In the forward phase, an error signal is computed from the input signal propagated through the network by fixing the values of the free parameters (the weights and biases). In the backward phase, the free parameters are adjusted to minimize the error signal.[25] In this work, we used feed-forward networks trained with BP technique.

Model evaluation methods

The model evaluation techniques are used in the algorithm to estimate the performance of the trained network and to determine if the network is accurate enough to be used in the optimization formulation. This section discusses two model evaluation techniques tested in our algorithm.

K-*fold cross-validation*

Cross-validation is a statistical technique that divides the sample data into subsets. In K-fold cross-validation, the data are equally split into K subsets or "folds." For each of these subsets, the $K - 1$ (K minus 1) folds are used as the training set to train the network and the one remaining set is used as the test set to test the performance of the trained network.[26] K different surrogate models are obtained in this technique. The surrogate-model deviation, σ_v, is calculated for each sample point of each test set K using Eq. 2. The maximum surrogate-model deviation, $\sigma_{\max}$, is the maximum value of all the surrogate-model deviations, as shown in Eq. 3.

$$\sigma_v = |y_v - \hat{y}_v|, \quad v \in [1, n] \tag{2}$$

$$\sigma_{\max} = \max_v \ \sigma_v, \quad v \in [1, n] \tag{3}$$

where n is the total number of sample points, y_v is the vth observed data point, and $\hat{y}_v$ is the corresponding prediction of the surrogate model for the vth data point.

Bootstrapping

In this technique, subsamples of the data are considered instead of dividing the data into subsets. In the application of model selection for ANNs, the original data set (B_o) is

used as the test set. The B bootstrapped bases (B_b*; $1 \leq b \leq B$) are generated by randomly sampling data points as replacements from the original data set (B_o). Each bootstrapped base with the same sample size (n) as B_o, B_b*, is used as the training set for constructing the network. Therefore, the bootstrapping technique provides B replications for the model.[27] B different surrogate models are obtained in this technique. Then, the surrogate-model deviation of each sample point in the test set B_b*, σ_v, is calculated using Eq. 4. The maximum value of all the surrogate-model deviations, σ_{max}, is calculated using Eq. 5.

$$\sigma_v = |y_v - y(x_v; \hat{\theta}^{*b})|, \quad v \in [1, n * B] \tag{4}$$

$$\sigma_{max} = \max_v \ \sigma_v, \quad v \in [1, n * B] \tag{5}$$

where $\hat{\theta}^{*b}$ is an estimator of the ANN parameter vector θ.

iLHS

LHS [28] is a stratified sampling method, which assures that all equal probability intervals are sampled. The LHS technique splits the range of each input variable into N equal intervals of probability $1/N$, where N is the number of sample points. Then, each of the N partitions is sampled once.[29] Because of the characteristics of LHS, the number of sample points is required in advance. It is generally not possible to reuse the existing sample points if it is concluded that more observations are needed at the end of the sampling. If additional sampling is performed, the final product will not remain a proper Latin hypercube pattern.[30]

The literature presents extensions of Latin hypercube sampling approach, which can increase the number of Latin hypercube sample points and yield a final Latin hypercube pattern by multiplying the original sample size (or base size) by sequential integer factors[31] and by doubling the previous sample size.[32,33] All of these methods are inflexible in term of the number of additional points. Yan and Minsker[34,35] study the estimation of the extension of Latin hypercube samples (N) by Q additional sample points. The sampling region is divided into $(N + Q)$ equal intervals. If there are more than Q empty intervals, the iLHS randomly selects Q empty intervals. Therefore, it is possible that some intervals may not be sampled. The extended sample size consists of the combination of all existing N sample points and Q additional sample points which are sampled from the randomly selected empty intervals. However, their algorithm is an approximation of the exact iLHS that does not guarantee the Latin hypercube pattern as the final product.

In this section, we explain our iLHS algorithm, which successfully reuses existing Latin hypercube sample points during the extension of the sample size. The algorithm extends the Latin hypercube samples by flexible number of additional sample points and yields the Latin hypercube pattern as the final product. The flowchart of the algorithm is shown in Figure 3.

Suppose that we have an original sample size of N points with P variables. An $N \times P$ matrix of original sample points is generated using LHS. If we want to increase the sample size by Q additional points, the iLHS starts by splitting the range of each decision variable into $(N + Q)$ equal probability intervals. These $(N + Q)$ equal probability intervals for each variable are then compared to the original N sample points for the same variable. There are three possible outcomes for each of the $P \times (N + Q)$ intervals: (1) empty intervals, the interval does not contain any original sample points; (2) full intervals, the interval contains a single sample point; or (3) overpopulated intervals, the interval contains more than one sample point.

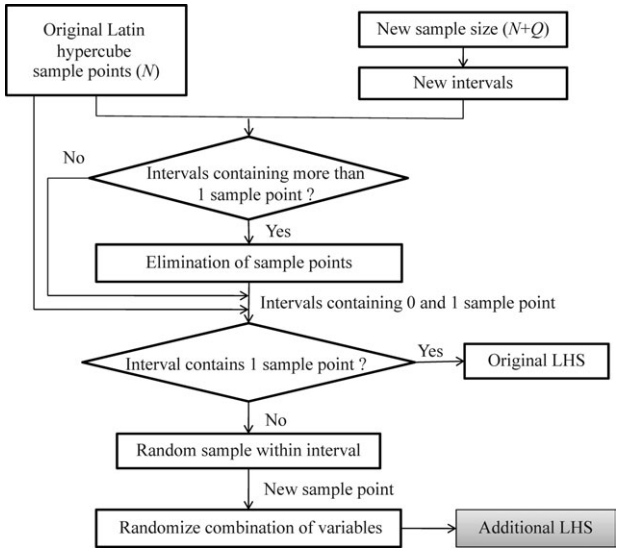

Figure 3. iLHS algorithm.

If there is at least one overpopulated interval, all but one of the data points in an overpopulated interval is eliminated by a two steps procedure. We have N sets of original sample points. Each set consists of P sample points that correspond to the variables. In the first step of the elimination procedure, we consider only the intervals that contain more than one sample point. To reuse the existing sample points as much as possible, we need to eliminate their sets as seldom as possible. Therefore, the set that fills one of the overpopulated intervals out of P variables is kept and the rest of the sets that fall into the same overpopulated interval are discarded. However, this approach does not guarantee that each overpopulated interval will have only one data set. In the second step, we search for the sets that contain a sample point in the same overpopulated interval and randomly keep one set and eliminate the others. Because the number of reusable sets, J, remaining from the original LHS sample points is less than the original sample points (N), the number of actual required additional experimental runs will be greater than the additional points (Q) needed to increase the sample size, but lower than the new sample size $(N + Q)$.

Following the elimination procedure, the reusable J sets are matched to the $P \times (N + Q)$ intervals again. There will be two categories of intervals at this time. The first category will contain a single sample point from the original LHS, and the other one will be empty. A new sample point is generated for each empty interval of each variable with random sampling within that interval, and the new sample points of variables are randomly combined to obtain the additional set of Latin hypercube sample points, a total of $N + Q - J \geq Q$ points.

Case Study: Optimization of CO_2 Capture Process with Aqueous Amines

The purpose of this case study, the optimization of a CO_2 capture process using surrogate models, that is, ANNs, to minimize the CO_2 capture cost, is to illustrate the application

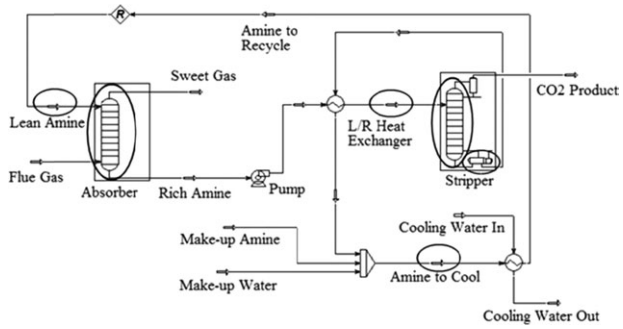

Figure 4. A schematic diagram of the conventional amine-based CO_2 capture plant.

of the developed sample size determination algorithm. Both operating and capital costs of CO_2 capture process were considered in this work.

The sample size determination algorithm is implemented with cross-validation and bootstrapping as the model evaluation techniques to compare the performance of both techniques. The value of the adjustable termination parameter, α, is determined by the comparison of the calculated slope ratio percentage with a visually stabilized curve of a statistical quantity. In what follows, we define the CO_2 capture process and the specifics of the case study.

Process simulation

1365 kg mol h^{-1} of gas turbine flue gas[8] is processed in a conventional amine-based absorption plant. The CO_2 capture process with a specification of 96 mol% CO_2 purity at the product stream is developed in Aspen HYSYS Version 7.1 (Figure 4). The amines property package using the Li-Mather electrolyte model is used as the thermodynamic model. The operating conditions and design variables, which are circled in Figure 4, are the solvent concentration, solvent circulation rate, reboiler duty, stripper feed temperature, and number of stages for the absorber and stripper columns. Both single and blended amines are studied to represent the problems with different numbers of decision variables. The studied solvents are monoethanolamine (MEA), diglycolamine (DGA), and blended MEA-methyldiethanolamine (MEA-MDEA). More details about process simulation and economic evaluation can be found in our previous work.[8]

Optimization using ANNs

For each decision variable of each amine solvent, the maximum and minimum values in Ref. 8 are used as the upper and the lower bounds, respectively (Table 1). We used a starting sample size of 10 times the number of the decision variables. Hence, we started with sample sizes of 60 for single amines (six decision variables) and 70 for blended amines (seven decision variables). LHS is performed to generate a test matrix of operating conditions and design variables. The Aspen HYSYS process simulation is run to obtain the output data, which are used in the economic analysis to estimate the CO_2 capture costs.

In the construction of the ANN, MATLAB Version 7.8.0 (R2009a) and Neural Network Fitting Tool (nftool) is used to train and test the ANNs. The data set of operating conditions and design variables are the input layer and the corresponding CO_2 capture costs are the output layer of the ANN.

Different ANN structures of one hidden layer with various numbers of neurons were trained. We only considered networks with one hidden layer because it has been shown that ANNs with one hidden layer are able to estimate any multivariable function with a limited number of discontinuities.[17] The considered number of neurons was between one and 20. The data set of the CO_2 capture process with aqueous MEA solvent was used to train and test the ANNs. For each sample size (60, 200, and 360) the data were randomly split into two groups: two-thirds of the original data was used as the training data set and the remaining one-third of the data was used as the testing data set. The ANN of one hidden layer with eight neurons resulted in lower values of sum of squared errors in all three sample sizes and hence this architecture was used throughout this work. The architecture of the ANN can also be determined with a mixed-integer nonlinear program (MINLP),[36] which uses binary variables (0 and 1) to represent the presence and the absence of the neurons, and the interconnections in the network. Given an upper bound on the prediction error between the output of the network and the actual output value, the number of neurons and/or the number of interactions in the network is minimized. Although the optimization approach would provide the best ANN architecture, given our focus on determining appropriate sample size for a selected ANN architecture in this work, our aforementioned approach is adequate to yield a good ANN architecture. Tangent sigmoid function (Eq. 7) and linear function (Eq. 6) are used as transfer functions for the hidden layer and the output layer, respectively.

Ten-folds, that is, $K = 10$, is used in the cross-validation approach because Kohavi[37] suggested that a 10-fold cross-validation is an effective method to use for model selection. For bootstrapping, Kallel et al.[27] advised that the value of B should be typically between 20 and 200. Given the small number of independent variables (decision variables in this study), we used $B = 20$ for this technique. To eliminate any bias that might result from slight changes in the original LHS or iLHS, the same data sets are used for both cross-validation and bootstrapping approaches. At each increment, the sample size is increased by one-third, β, of the current

Table 1. The Upper and Lower Bounds of the Operating Conditions and Design Variables for Each Amine Solvent

Solvents	UB/LB	Amine Conc. (wt%) 1°	Amine Conc. (wt%) 3°	Absorber (stages)	Stripper (stages)	Cir. Rate (m^3/h)	Reboiler Duty (kW)	Regenerator-Inlet Temp. (°C)
MEA	UB	24	–	24	12	62	5338	105
	LB	10	–	8	7	43	3248	81
DGA	UB	62	–	23	12	35	4225	105
	LB	48	–	8	7	15	1379	88
Blended 1	UB	22	22	24	12	59	5338	105
	LB	7	13	8	7	43	3078	83

UB, LB, and Blended 1 correspond to upper bound, lower bound and MEA-MDEA, respectively.

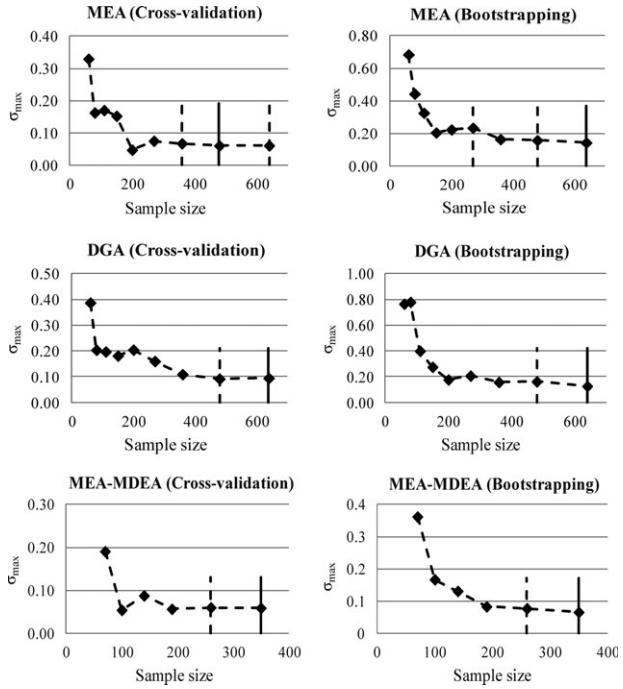

Figure 5. The plots of the maximum surrogate-model deviation versus sample size for both model evaluation techniques.

number of sample points and then rounded up to the nearest tenth. This was done to ensure that all of the 10-folds in the cross-validation had an equal number of sample points.

Given the data from the appropriate initial sample size algorithm (''ANNs'' Section), the ANN is built that represents the relationship between the inputs and the outputs. The ANN architecture specifics, the weights, biases and the transfer functions of the neurons are incorporated into the optimization problem. The nonconvex MINLP with the ANN is formulated in GAMS Version 23.5 and solved using BARON Version 9.0.6 to obtain the solution. In the MINLP formulation, the design parameters which are the number of absorber and stripper stages are integer variables and the operating conditions are continuous variables. The formulation which minimizes the CO_2 capture cost is given by:

$$\begin{aligned} &\text{Decision variables} : x_m \quad \forall m \in \{1,2,...,M\} \\ &\text{MIN}: \quad y = \left(\sum\nolimits_s W2_s \times L1_s\right) + C2 \end{aligned} \tag{6}$$

$$\begin{aligned} \text{Subject to} : L1_s &= \frac{2}{1+e^{-2(\Sigma m(W1_{sm}\times x_m)+C1_s)}} - 1 \\ x_m &\in \mathcal{Z} \;\forall\; m \in \{1,2\} \\ x_{m,LB} &\le x_m \le x_{m,UB} \end{aligned} \tag{7}$$

where M is the total number of decision variables. W is the input weight and C is the input bias. Numbers *1* and *2* are the first layer (hidden layer) and the output layer, respectively. The index m represents the independent (i.e., decision) variables in the input layer, and s represents the neurons in the hidden layer. Decision variables 1 and 2 are the number of absorber stages and the number of stripper stages, respectively.

Results and Discussion

For all aqueous amine solvents, the comparisons between the visual and calculated results for the sample size determination are given. The visual curves show the plots of the maximum surrogate-model deviation versus the sample size for both model evaluation techniques for all amines are shown in Figure 5. A dashed vertical line in Figure 5 indicates the sample size where $\sigma_{\max}$ shows a small change between two sample sizes (the slope ratio percentage is less than 2%), and a solid vertical line on each graph indicates the sample size where $\sigma_{\max}$ stabilizes and the algorithm is terminated.

The slope ratio percentages obtained by the sample size determination algorithm of all amine solvents are shown in Table 2. As explained before, the slope ratio percentage is used to determine the stabilization of $\sigma_{\max}$. The slope ratio percentage calculation can be seen in Eq. 8.

$$\text{Slope ratio}_{a_{\sigma,i}} = \frac{\frac{|\sigma_{\max_i} - \sigma_{\max_i - 1}|}{n_i - n_{i-1}}}{a_{\sigma,\text{highest}}} \times 100 \tag{8}$$

When we consider both Figure 5 and Table 2, we can see that all of the slope ratio percentages of less than 2% are compatible with the visually stabilized curves.

Table 3 show the optimal operating conditions and design variables obtained using the generated ANNs for the CO_2 capture process with aqueous MEA, DGA, and blended MEA-MDEA. The tables also shows the percent errors of the CO_2 costs predicted by the network constructed with each sample size. We refer to the objective function value at the solution obtained using the surrogate models as the predicted output. The actual output, that is, the one that is calculated by the process simulator, is obtained by running the process simulator at the solution obtained using the ANN.

Table 2. The Calculated Slope Ratio Percentage of $\lambda_{\max}$ for the CO_2 Capture Process with Each Amine Solvent

		Slope Ratio (%)	
Solvents	Sample Size	Cross-Validation	Bootstrapping
MEA	60	–	–
	80	100.00	100.00
	110	2.98	32.43
	150	5.30	25.11
	200	25.24	3.07
	270	4.75	**1.26**
	360	**1.03**	6.45
	480	**0.68**	**0.42**
	640	**0.01**	**0.76**
DGA	60	–	–
	80	100.00	100.00
	110	2.37	100.00
	150	4.34	24.53
	200	5.02	15.51
	270	6.88	3.39
	360	6.17	4.35
	480	**1.58**	**0.32**
	640	**0.18**	**1.76**
MEA-MDEA	70	–	–
	100	100.00	100.00
	140	18.46	13.82
	190	13.71	14.80
	260	**1.09**	**1.32**
	350	**0.30**	**1.95**

The bold values represent the slope ratio percentages of the maximum surrogate-model deviations of less than 2%.

Table 3. The Operating Conditions and Design Variables Results of the CO_2 Capture Process

Solvents	Sample Size	Total Simulation Runs	Operating Conditions and Design Variables: 1a (wt%)	1b (wt%)	2 (stages)	3 (stages)	4 (m^3/h)	5 (kW)	6 (°C)	CO_2 Capture Cost ($/ton CO_2 recovered): GAMS	HYSYS	Percent Error (%)
MEA	270	475	12.28	–	22	10	61.98	3249	81.00	48.45	57.31	15.47
	480	880	12.52	–	22	12	57.57	3356	81.04	48.55	48.98	0.89
	640	1192	12.75	–	21	12	58.74	3441	81.04	48.79	48.81	0.05
DGA	*640*	1216	48.01	–	23	12	22.67	1582	102.70	42.71	42.97	0.61
MEA-MDEA	*350*	633	9.74	13.01	21	12	58.96	3365	83.03	48.22	48.42	0.40

The sample sizes where the algorithm terminates are shown in italic.

Equation 9 shows the calculation of the percent error between the predicted output and the actual output:

$$\text{Percent error} = \frac{|\text{Predicted value} - \text{Actual value}|}{\text{Actual value}} \times 100 \quad (9)$$

In Table 3, the numbers at the top correspond to the operating conditions and design variables; 1a represents the MEA or DEA solvent concentration, 1b represents the MDEA solvent concentration, 2 represents the number of stages for the absorber column, 3 represents the number of stages for the stripper column, 4 represents the solvent circulation rate, 5 represents the reboiler duty, and 6 represents the stripper feed temperature.

As can be seen in Figure 5, the value of σ_{max} might stay nearly the same for two consecutive times (by chance) yielding a small slope ratio percentage between these two sample sizes. However, it is statistically very unlikely for the value of σ_{max} to be nearly the same for three consecutive times unless σ_{max} has actually stabilized. For example, using bootstrapping as the model evaluation technique, the slope ratio percentage is below 2% (our preset termination criterion) for sample size of 270 in case of aqueous MEA solvent. If this sample size is used to construct the ANN for the optimization problem, the percent error between the predicted output and the actual output turns out to be 15.47%, which is not accurate. The sample size obtained when the algorithm terminates is 640. The ANN constructed with sample size of 640 yields the percent error of 0.05%. For the aqueous MEA solvent, the algorithm terminates at the sample size of 480 with the cross-validation approach. The ANN constructed with sample size of 480 yields 0.89% as the percent error. Furthermore, with the cross-validation approach, the third slope ratio percentage is also under 2%, which supports our conclusion. Hence, we recommend that at least two consecutive slope ratio percentages stay below the preset termination value, α, for the determination of the appropriate sample size.

In case of aqueous DGA solvent, the algorithm terminates at the sample size of 640 for both cross-validation and bootstrapping. The ANN with sample sizes of 640 was constructed and used as the objective function in the optimization problem. As we can see from Table 3, the percent error between the predicted CO_2 capture costs from the network and the actual CO_2 capture cost (obtained from HYSYS simulation) is 0.61%. For blended MEA-MDEA, the algorithm terminates at the sample size of 350 for both cross-validation and bootstrapping. The result obtained from the ANN constructed with the sample size of 350 shows the value of percent error of 0.40%. The information of the ANN (constructed with the sample size mentioned in this section using the normalized inputs and outputs in the range of $[-1, 1]$) for each aqueous amine solvent can be found in Supporting Information. This information includes the mean squared errors of the training set, validation set, and test set, and the weights and biases of the network.

The case study of aqueous MEA reveals that two consecutive slope ratio percentages are necessary and sufficient to indicate the stabilization of σ_{max}. All case studies show that both model evaluation approaches (cross-validation and bootstrapping) are reliable to be used in the algorithm. The algorithm is able to determine the appropriate sample size to construct an accurate ANN which yields a percent error of less than 1% for the objective function value.

Conclusions

In this article, an algorithm to determine the appropriate sample size to construct an accurate surrogate model to be used in optimization problems is presented. The proposed sample size determination algorithm was successfully used in the optimization of a CO_2 capture process with aqueous amines. In the algorithm, the model evaluation methods are used to estimate the performance of various networks constructed with different sample sizes. Our analyses show that both cross-validation and bootstrapping are suitable to be used as the model evaluation methods. LHS was used as a sampling technique to evenly spread the sampling points over the entire sample space. However, LHS can be replaced with any surface-covering sampling technique. There are some disadvantages of LHS, such as the fact that additional sample points cannot be added to an existing Latin hypercube sample in the case of extended sample sizes. In this case, iLHS was developed to reuse existing Latin hypercube sample points for extended sample sizes. The iLHS algorithm successfully reduced the number of required simulation runs for the case study.

For future work, the disaggregation–aggregation-based approaches[22] will be incorporated into the sample size determination algorithm to compare the computational benefits. The disaggregation–aggregation based approaches can be applied to two main parts of the algorithm to deal with large sample sizes. The first part is the data collection. In this part, the total number of sample points can be split into subsets and then simulated using parallel computers. The second part is the model evaluation. From the overall data, several subsets containing both training data and test data sets can be provided. The number of subsets and the procedures differ according to the model evaluation technique. Then, the ANNs for each of the subsets can be trained and evaluated using parallel computers to give the maximum surrogate deviation. After that, the values of each maximum surrogate deviation obtained from each computer can be compared for the highest value to be used in the algorithm.

Acknowledgment

The financial support from the University of Tulsa is gratefully acknowledged.

Literature Cited

1. Nishida N, Stephanopoulos G, Westerberg AW. A review of process synthesis. *AIChE J*. 1981;27:321–351.
2. Yeomans H, Grossmann IE. A systematic modeling framework of superstructure optimization in process synthesis. *Comput Chem Eng*. 1999;23:709–731.
3. Qian Z, Conner Seepersad C, Joseph Roshan V, Allen JK, Jeff Wu CF. Building surrogate models based on detailed and approximate simulations. *J Mech Des*. 2006;128:668–677.
4. Wan X, Pekny JF, Reklaitis GV. Simulation-based optimization with surrogate models—application to supply chain management. *Comput Chem Eng*. 2005;29:1317–1328.
5. Hasan SDM, Melo DNC, Filho RM. Simulation and response surface analysis for the optimization of a three-phase catalytic slurry reactor. *Chem Eng Process*. 2005;44:335–343.
6. Wang H, Li C, Li C. RSM optimization of the operating parameters for a butanol distillation column. *Asia-Pacific J Chem Eng*. 2010;7:117–123.
7. Mariano AP, Costa CBB, de Angelis DD, Filho FM, Atala DIP, Maciel MRW, Filho RM. Optimisation of a continuous flash fermentation for butanol production using the response surface methodology. *Chem Eng Res Des*. 2010;88:562–571.
8. Nuchitprasittichai A, Cremaschi S. Optimization of CO_2 capture process with aqueous amines using response surface methodology. *Comput Chem Eng*. 2011;35:1521–1531.
9. Lefèvre S, Henry Ferrasse J, Boutin O, Sergent M, Faucherand R, Viand A. Process optimisation using the combination of simulation and experimental design approach: application to wet air oxidation. *Chem Eng Res Des*. 2011;89:1045–1055.
10. Simpson TW, Mauery TM, Korte JJ, Mistree F. Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA J*. 2001;39:2233–2241.
11. Caballero JA, Grossmann IE. An algorithm for the use of surrogate models in modular flowsheet optimization. *AIChE J*. 2008;54:2633–2650.
12. Davis E, Ierapetritou M. A kriging method for the solution of nonlinear programs with black-box functions. *AIChE J*. 2007;53:2001–2012.
13. Fernandes FAN. Optimization of Fischer-Tropsch synthesis using neural networks. *Chem Eng Technol*. 2006;29:449–453.
14. Basheer IA, Hajmeer M. Artificial neural networks: fundamentals, computing, design, and application. *J Microbiol Methods*. 2000;43:3–31.
15. Benardos PG, Vosniakos GC. Optimizing feedforward artificial neural network architecture. *Eng Appl Artif Intell*. 2007;20:365–382.
16. Henao CA, Maravelias CT. Surrogate-based process synthesis. 20th European Symposium on Computer Aided Process Engineering, Ischia, Italy, 2010;1129–1134.
17. Henao CA, Maravelias CT. Surrogate-based superstructure optimization framework. *AIChE J*. 2011;57:1216–1232.
18. Nascimento CAO, Giudici R, Guardani R. Neural network based approach for optimization of industrial chemical processes. *Comput Chem Eng*. 2000;24:2303–2314.
19. Yuste AJ, Dorado MP. A neural network approach to simulate biodiesel production from waste olive oil. *Energy Fuels*. 2006;20:399–402.
20. Satyanarayana A, Davidson I. A dynamic adaptive sampling algorithm (DASA) for real world applications: finger print recognition and face recognition. *Found Intell Syst*. 2005;3488:5–25.
21. Devabhaktuni VK, Zhang QJ. Neural network training-driven adaptive sampling algorithm for microwave modeling. In: Proceedings of 30th European Microwave Conference. Paris, France, 2000;222–225.
22. Gueddar T, Dua V. Disaggregation–aggregation based model reduction for refinery-wide optimization. *Comput Chem Eng*. 2011;35:1838–1856.
23. Dhabak D, Pandit S. Adaptive sampling algorithm for ANN-based performance modeling of nano-scale CMOS inverter. *World Acad Sci Eng Technol*. 2011;80:812–818.
24. Hayati M, Shirvany Y. Artificial neural network approach for short term load forecasting for illam region. *World Acad Sci Eng Technol*. 2007;28:280–284.
25. Haykin S. *Neural Networks and Learning Machines*. New York: Prentice Hall, 2009.
26. Forrester AIJ, Keane AJ. Recent advances in surrogate-based optimization. *Prog Aerosp Sci*. 2009;45:50–79.
27. Kallel R, Cottrell M, Vigneron V. Bootstrap for neural model selection. *Neurocomputing*. 2007;48:1–4.
28. McKay MD, Beckman RJ, Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*. 1979;21:239–245.
29. Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Tucker PK. Surrogate-based analysis and optimization. *Prog Aerosp Sci*. 2005;41:1–28.
30. Vamvatsikos D. Estimating seismic performance uncertainty using IDA with progressive accelerogram-wise Latin hypercube sampling. In: Proceedings of the 11th International Conference on Applications of Statistics and Probability in Civil Engineering. Zurich, 2011.
31. Pleming JB, Manteufel RD. Replicated Latin hypercube sampling. In: Proceedings of 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 16-21 April 2005, Austin (TX), 2005.
32. Sallaberry CJ, Helton JC, Hora SC. Extension of Latin hypercube samples with correlated variables. *Reliab Eng Syst Saf*. 2008;93:1047–1059.
33. Vamvatsikos D, Lignos DG. Evaluating the epistemic uncertainty of the seismic demand and capacity for a 9-story steel moment-resisting frame. In: Proceedings of the 7th Hellenic National Conference on Steel Structures, Volos, Greece, 2011.
34. Yan S. Optimizing groundwater remediation designs using dynamic meta-models and genetic algorithms. 2006, PhD Thesis, University of Illinois at Urbana-Champaign, IL, 2006.
35. Yan S, Minsker B. Applying dynamic surrogate models in noisy genetic algorithms to optimize groundwater remediation designs. *J Water Resour Plann Manage*. 2011;137:284–292.
36. Dua V. A mixed-integer programming approach for optimal configuration of artificial neural networks. *Chem Eng Res Des*. 2010;88:55–60.
37. Kohavi R. A Study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the 14th International Joint Conference on Artificial intelligence, Montréal, Québec, Canada. 1995;1137–1143.

Manuscript received Nov. 29, 2011, and revision received May 4, 2012.