Prediction of the Trayed Distillation Column Mass-Transfer Performance by Neural Networks

Eric Olivier and R. Bruce Eldridge*

The Separations Research Program, Mail Code R7100, The University of Texas at Austin, Austin, Texas 78758

The sieve-tray distillation column mass-transfer efficiency was successfully modeled using a neural network. The database developed by Garcia and Fair (*Ind. Eng. Chem. Res.* **2000**, *39*, 1809) was utilized to train and validate the neural network model. The results indicate that, if the system is similar to the data used to train the neural network, the purely empirical neural network model yields very accurate predictions. However, in areas where data are lacking, even if the input parameters are in the same range as those in the database, the neural network model must make extrapolations and will therefore make unreliable predictions. MS Excel programs based on the results from neural networks were developed for the prediction of the sieve-tray efficiency and structured packing height equivalent to a theoretical plate. These programs represent a convenient and easy to use tool. Moreover, they make better predictions than a single neural network and make it possible to detect extrapolations and misleading predictions.

Introduction

Overview. In a previous work,[2] the height equivalent to a theoretical plate (HETP) of nine different types of structured packings was successfully modeled via a neural network. These results indicated that a neural network is an effective alternative to fundamental and semiempirical models. The modeling of the sieve-tray point efficiency represents a problem as complex as the modeling of structured packing HETP. Similarly, the sieve-tray point efficiency depends on the geometry of the separation device, the physical properties of the chemical system, and the operating conditions of the column. For both, fundamental models, involving mass-transfer and hydrodynamics considerations, have been developed. A large database is available for the sieve-tray point efficiency which can be used to both train and validate the neural network approach. The neural network modeling of HETP and sieve-tray efficiency requires significant manipulations of the databases and the use of software which is not readily available. Consequently, to facilitate the use of the neural network methodology, the neural network was converted to a universally usable MS Excel format.

Neural Network Fundamentals. A neural network is a structure, in which each node performs computations (defined by the activation function) and each connection conveys a signal from one node to another labeled by a "weight", that indicates the extent to which the signal is amplified or diminished by a connection.

In the current study, neural networks are used to fit a set of experimental points in order to provide a purely empirical model. The experimental points are called the training cases (or learning cases) and form the training set (or learning set). They consist of input vectors (values of the input variables) associated with the experimental output value. For a given neural network structure and a given activation function, the fitting procedure consists of minimizing a target function defined as the distance between the outputs of the neural network and the experimental outputs, given the inputs in the training set. The target function is a function of the weights. This procedure is called the training phase. The fitting procedure is carried out by an algorithm, which, step by step, iteration after iteration, will "lead" from an initial set of weights to the set of weights where the target function reaches its minimum.

During the training phase, the outputs of a neural network come to approximate the output values given the inputs in the training set. This ability may be useful in itself, but more often the purpose of using a neural network is to generalize. The generalizability of a neural network is its ability to make accurate predictions for cases that are not in the training set. Minimizing the target function does not necessarily mean improving the ability of the neural network to generalize. Consequently, the optimization does not consist of finding the minimum of the target function but of finding the neural network structure and the values of the weights that provide the best generalization. After the neural network is optimized, its structure and its weights are frozen and the neural network can be used to predict the output value corresponding to cases that are not in the training case.

There are several different types of structures for neural networks. Feedforward neural networks are the most popular and were used in the current work.

To illustrate a feedforward neural network, consider the simplified network structure shown in Figure 1.

This neural network is a function in the two-dimensional space. There are two input variables: x_1 and x_2. The output of the function is a scalar, z.

There are five nodes: two input nodes, two hidden nodes (a hidden node is any node that is neither an input node nor an output node), and one output node.

* Corresponding author. E-mail: rbeldr@che.utexas.edu. Phone: 512-471-7067. Fax: 512-471-1720.

10.1021/ie010499l CCC: $22.00
Published on Web 04/20/2002

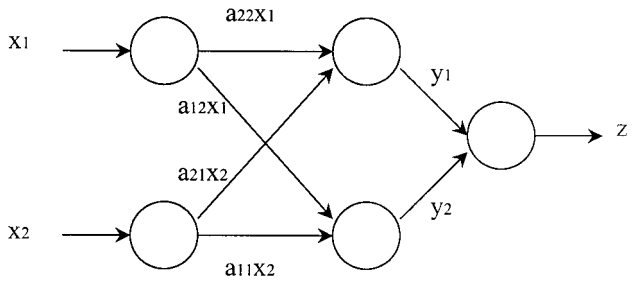

Figure 1. Example of a feedforward neural network.

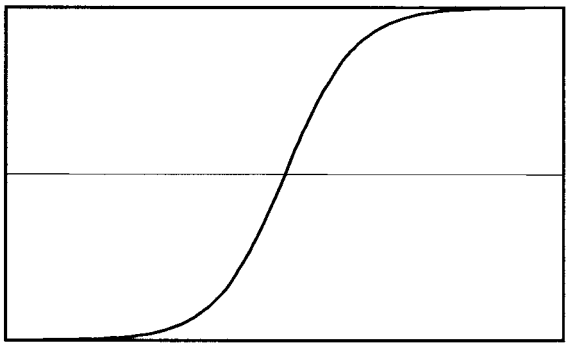

Figure 2. Sigmoidal function [f(net) = tanh(net)].

The output value z is expressed by

$$z = b_0 + b_1x_1 + b_2x_2$$

and

$$y_1 = f(a_{01} + a_{11}x_1 + a_{21}x_2)$$

$$y_2 = f(a_{02} + a_{12}x_1 + a_{22}x_2)$$

where f is the activation function. a_{11}, a_{21}, a_{12}, and a_{22} are the weights of the first layer and b_1 and b_2 the weights of the second layer. a_{01} and a_{02} are the biases of the first layer, and b_0 is the bias of the second layer. To simplify, weights and biases will both be called weights.

The most popular activation functions are "sigmoid" (S-shaped) functions, whose output is illustrated in Figure 2. These functions are continuous and differentiable everywhere, are rotationally symmetric about some point, and asymptotically approach their saturation values.

Some choices for f are

$$f(\text{net}) = z + \frac{1}{1 + \exp[-x(\text{net}) + y]}$$

and

$$f(\text{net}) = \tanh[x(\text{net}) - y] + z$$

These functions facilitate the training of the neural network.

Neural Network Optimization. Consider a training set $\{(x_p, z_p), p = 1, ..., P\}$, with P representing training cases and every case defined by x_p, its input vector (size n), and z_p, its output value. Next consider a neural network that is a function of n variables. Its structure and its activation function are fixed. Therefore, the neural network is entirely defined by its weights. If the input of the neural network is x_p, its output value is o_p. The goal of training is to modify the weights of the neural network so that the output value o_p is as close as possible to the actual value z_p. Toward this goal, the cumulative error or distance function D is minimized.

$$D = \sum_{p=1}^{P} \text{Err}(z_p, o_p)$$

Err is a norm (for example, Euclidean distance or Manhattan distance).

The training phase starts with randomly chosen initial weight values. Then an optimization algorithm is applied: after each iteration, the weights are modified so that the cumulative error decreases. There are a variety of algorithms. The back-propagation algorithm was used in this study. In back-propagation, the weight changes are proportional to the negative gradient of the error. More details about learning algorithms are available elsewhere.[3]

Training the neural network is necessary to later make predictions for cases that were not in the training set: if the neural network is not trained adequately, i.e., if the fit on training cases is not accurate enough, the prediction on testing cases (fresh cases not used during the training phase) will be poor (see Figure 3). In this case, the neural network model suffers from underfitting. However, a good fit does not guarantee a good generalizability: the error on the training cases may be very low, but errors on test cases may be high (consider test points between the training points on the overfitting graph). In this situation, the model suffers from overfitting. Overfitting may occur when the number of weights (that is related to the number of nodes) is too high relative to the number of training cases or if the number of training iterations is too high ("epochs").

Optimizing a neural network does not only consist of minimizing the cumulative error. Because the purpose of the neural network is to generalize, the structure (number of layers and number of hidden nodes) and the training time have to be optimized. Therefore, a tradeoff must be found between underfitting and overfitting. Optimizing a neural network consists of determining the structure and the weights that provide the best generalization. Some books and articles offer rules of thumb for choosing the structure (number of layers and number of hidden nodes). However, these rules ignore the number of training cases and the complexity of the function to be modeled. Therefore, the best way to optimize structure and weights is to evaluate the generalization error of different neural networks. There are different methods to estimate the generalization error. One method is the split-sample or hold-out validation: one part of the data is reserved as a test set, which must not be used during training. After training, the network is run on the test set, and the error on the test set provides an estimate of the generalization error. The disadvantage of split-sample validation is that it reduces the amount of data available for both training and validation.

Another method is the cross-validation. Cross-validation is an improvement on split-sample validation that allows you to use all of the data for training. The database is divided into k subsets of equal size. The neural network is trained k times, each time leaving out one of the subsets from training but using only the omitted subset to evaluate the generalization error. The disadvantage of cross-validation is that you have to retrain the net many times.

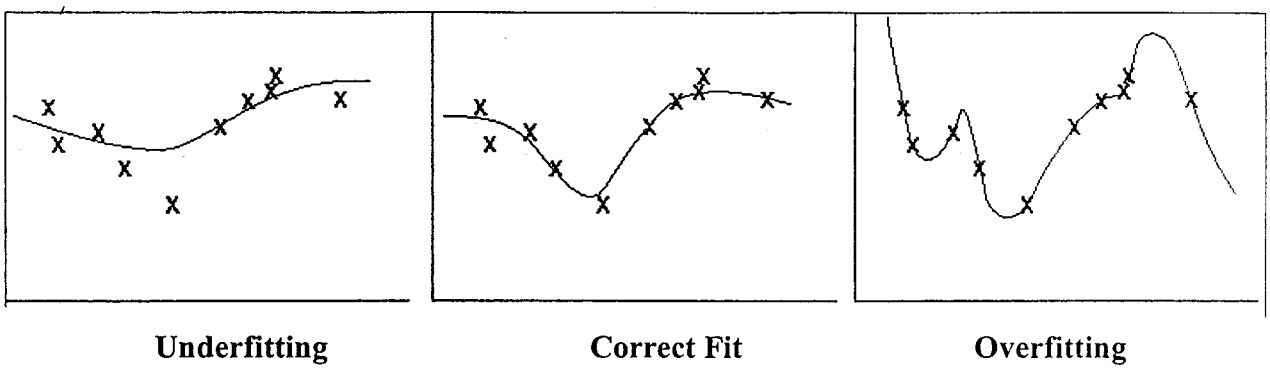

Figure 3. Underfitting and overfitting.

For a particular structure (i.e., for a certain number of hidden layers and hidden nodes), the values of the weights and so the performance of the neural network depend on the training time. So, to get the optimized weights, we must determine the optimal training time.

In the current study, the systematic method for the optimization was used: several neural networks with different structures,were trained, using different training times. Then, the generalizability of every neural network was evaluated by cross-validation. Therefore, the structure and training time that provide the best generalization were obtained.

Sieve-Tray Operation. Because of its simplicity of design and the effectiveness of contacting, the cross-flow sieve tray is widely used for commercial distillation columns. Its purpose is to promote intimate contacting of the phases and thus to encourage rapid transfer of material between the phases. Basically, a sieve tray consists of perforated plates. The liquid passes horizontally onto the plate ("cross-flow") and then flows down to the lower plate, whereas the vapor passes vertically through the holes. Therefore, a vapor phase is dispersed in the liquid phase, promoting liquid/vapor contact.

Two different contacting regimes can occur on a cross-flow sieve tray: the jetting regime and the bubbling regime. In the jetting regime, a continuous jet is formed, breaking up into bubbles of different sizes. In the bubbling regime, bubbles are formed at the holes, with a bimodal distribution ("large" bubbles and "small" bubbles). These regimes can occur simultaneously: the contacting regime on the tray ranges from a vapor-continuous "spray" at high vapor/liquid ratios to "froth" at lower ratios.

Mass-Transfer Efficiency. The design of a distillation column starts with the determination of theoretical plates necessary for the given separation. Then, by using a column efficiency, the actual number of plates is deduced. The column efficiency is calculated from the overall tray efficiency, and the overall tray efficiency is calculated from the point efficiency. Consequently, the point efficiency is a key parameter for the design of sieve-tray distillation columns. The three different efficiencies (point efficiency, overall tray efficiency, and column efficiency) are defined below.

(i) Point Efficiency (or Local Efficiency). The point efficiency of tray n is expressed by

$$E_{OG} = \left(\frac{y_n - y_{n-1}}{y_n^* - y_{n-1}}\right)_{point}$$

Considering a mass balance across a differential element in the froth of a sieve tray, the vapor and liquid phase mass-transfer units are expressed by

$$N_G = k_G a_i' t_G$$

$$N_L = k_L a_i' t_L$$

The number of overall transfer units, based on gas concentrations, is expressed by

$$\frac{1}{N_{OG}} = \frac{1}{N_G} + \frac{\lambda}{N_L}$$

Then, the value of the point efficiency is computed from N_{OG}:

$$E_{OG} = 1 - \exp(1 - N_{OG})$$

(ii) Overall Tray Efficiency (or Murphree Efficiency). The overall tray efficiency, E_{MV}, is expressed by

$$E_{MV} = [\exp(\lambda E_{OG}) - 1]/\lambda$$

where E_{OG} is the point efficiency.

(iii) Column Efficiency. This efficiency is simply the ratio of the number of theoretical plates to the actual number of plates required for a given separation. The column efficiency is deduced from the overall tray efficiency. At total reflux, the value of the column efficiency is very close to the value of the overall tray efficiency. For finite reflux, the column efficiency may be deduced from the overall efficiency by the relationship developed by Lewis.[4]

Current Models for the Sieve-Tray Point Efficiency. Both the number of theoretical plates and the column efficiency are necessary for the design of a sieve-tray column. The prediction of theoretical plates is highly developed and based on rigorous computer models. However, relatively little attention has been given to the prediction of the tray efficiency. The column efficiency is deduced from the point efficiency. The determination of the point efficiency requires the knowledge of the interfacial surface area, liquid and vapor residence times, and liquid- and gas-phase mass-transfer coefficients. Several models have been proposed to describe the sieve-tray efficiency: Chan[5] and Locket.[6] All of these models are semiempirical and do not take into account the two regimes that prevail on a cross-flow tray ("spray" and "bubbling" regimes). Prado and Fair were the first to develop a purely theoretical model for the prediction of the point efficiency for a water−air system.[7] This model has been modified and extended to hydrocarbon systems by Garcia and Fair.[8] The Garcia

Table 1. Results of the 9 × 10 Cross-Validation Optimization

no. of hidden nodes	no. of epochs	run 1	2	3	4	5	6	7	8	9	10	absolute average epochs
	150	5.94	4.10	5.45	4.88	6.94	5.05	5.07	7.47	7.18	6.67	5.87
		5.30	10.33	8.89	6.05	6.92	9.36	8.83	9.56	12.31	11.99	8.96
11	200	5.23	5.86	3.61	3.56	3.18	4.46	5.49	3.77	3.99	3.89	4.30
		4.17	11.94	7.37	5.14	8.12	11	10.40	8.17	11.15	8.86	8.63
	250	5.84	5.56	4.35	4.98	6.46	7.20	7.02	5.65	3.75	5.24	5.60
		5.42	12.34	8.18	6.43	7.27	13.7	8.43	8.87	8.25	13.93	9.28
	150	4.51	5.28	4.23	5.13	6.51	7.44	4.03	3.95	4.77	3.86	4.97
		5.42	12.25	7.5	6.22	9.33	9.54	8.15	6.60	9.66	8.38	8.30
12	200	3.56	4.48	4.60	3.12	5.15	5.13	4.37	5.39	3.56	3.67	***4.30***
		4.91	11.29	6.77	6.43	6.95	8.34	7.83	6.9	8.49	9.98	***7.79***
	250	5.66	4.29	4.04	4.50	4.04	3.6	4.35	6.26	4.28	3.55	4.46
		4.24	12.26	5.65	6.00	6.28	9.94	6.41	9.14	11.13	11.04	8.21
	100	5.14	6.84	6.52	5.18	6.84	4.43	5.69	5.98	7.71	4.35	5.87
		4.68	12.17	7.65	6.64	6.02	10.4	7.65	10.13	13.94	8.36	8.76
13	150	4.41	5.74	4.46	5.26	6.37	5.00	5.56	3.99	5.02	4.91	5.07
		5.75	12.40	5.43	7.15	7.55	6.76	13.00	6.72	8.30	7.59	8.06
	200	5.68	2.91	5.27	5.92	4.34	7.92	5.39	3.15	3.51	3.44	4.75
		5.32	9.89	6.13	5.97	6.75	10.90	9.45	6.30	11.83	12.49	8.50

and Fair model has been tested on the database they developed. A total of 87% of the predicted efficiencies were found to be within ±25% band, and the absolute average error was equal to 18.22%.[8]

Previous Neural Network Mass-Transfer Model Development. In a previous work,[2] the structured packing HETP was modeled via a neural network. The network was trained on a 445-point database and used 15 input parameters describing the physical properties of the system, the structure of the packing, and the operating conditions. In the range of input parameters used to train the network, the purely empirical model yielded very accurate predictions which were more accurate than semiempirical and fundamental models.[9,10] Motivated by the prediction of the purely empirical model for the prediction of structured packing HETP, the applicability of neural network technology for the prediction of the sieve-tray point efficiency was conducted. A large database[1] (279 experimental points) is available which contained a variety of parameters describing the physical properties of the system, the geometry of the tray, and the operating conditions.

Development of the Neural Network

Database. The first step in neural network modeling is the development of a database. The database is necessary to train the network and to estimate its ability to generalize. A database for the sieve-tray point efficiency was previously developed by Garcia and Fair.[1] This database combines data taken on a semiindustrial scale, published data, and data released by Fractionation Research Incorporated. The database offers a large scope of points (279 experimental cases), including 13 chemical systems, 11 tray geometries, and a large range of liquid and gas velocities (see the appendix). Each experimental case consists of the measured sieve-tray point efficiency and the related values of the parameters describing the chemical system, the operating conditions, and the geometry of the tray. The parameters were known (geometry of the tray), measured (composition, flow rates, temperature, and pressure), or calculated with ASPENPLUS (physical properties via temperature, pressure, and composition).

Choice of Input Parameters. Even if a neural network is a purely empirical model, the understanding of the physical mechanisms involved on a sieve tray is necessary. In fact, the inputs to the network need to contain sufficient information pertaining to the target, so that there exists a mathematical function relating correct outputs to inputs with the desired degree of accuracy.

The fundamental model developed by Prado and Fair reveals that the sieve-tray point efficiency is entirely defined by 13 variables: eight parameters describing the physical properties of the system (ρ_L, ρ_V, μ_L, μ_V, D_L, D_V, σ, and m), three parameters describing the geometry of the tray (d_H, h_w, and A_F), and two parameters describing the flow regime (F_{AA} and Q_{lw}).

From the experimental cases, the training cases were built: a training case is the measured efficiency associated with the 13 input parameters presented above.

Optimization of the Structure and the Training Time. All training cases were combined into a master spreadsheet. All input variables for each data point were normalized to a similar magnitude. Normalization can make training faster and reduce the chances of getting stuck in local optima. If x_{ij} is the ith input parameter of the jth point in the training set, X_{ij}, the normalized value of x_{ij} is defined as

$$X_{ij} = (x_{ij} - \bar{x}_i)/\sigma_i$$

where $\bar{x}_i$ is the average of the ith parameter on all of the training sets and σ_i its standard deviation. Initially, the performance of the simplest neural network was studied: a feedforward neural network with only one hidden layer. The activation function is a hyperbolic tangent.

The training time and the number of hidden nodes were optimized using the "9 × 10 cross-correlation": the data were randomly divided 10 times into training sets (90% of the data) and test sets (10% of the data). For each combination training time/number of hidden nodes ("run"), the neural network was trained on the 10 different training sets and tested on the 10 corresponding test sets. The weights were optimized using the back-propagation algorithm. The performance of the neural networks on training and test sets was evaluated by the absolute average error. The results are reported in Table 1. For every combination hidden nodes/training time, the absolute average error on the train set is reported in the first row and that on the test set in the second row.

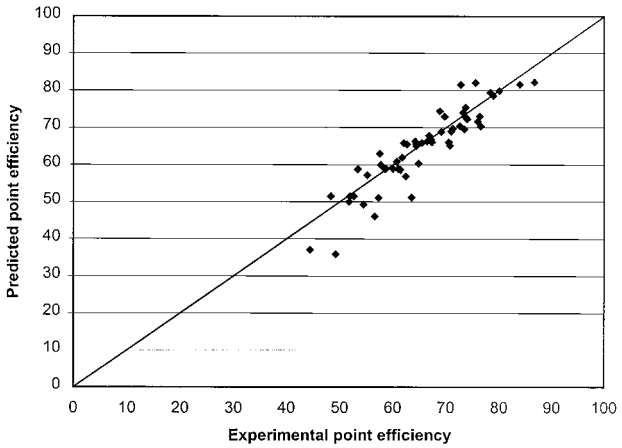

Figure 4. Parity plot for test data set (without any training cases).

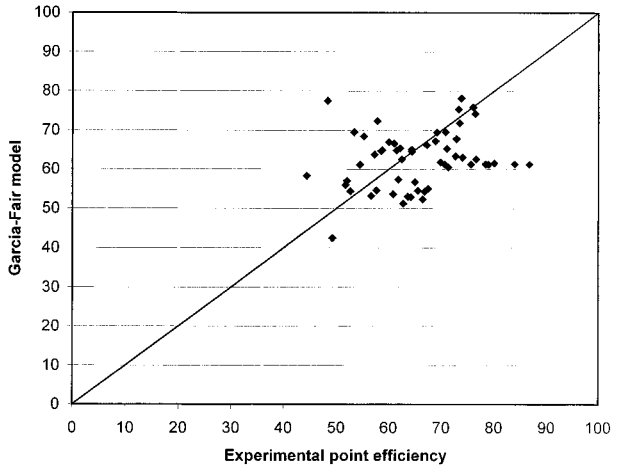

Figure 5. Parity plot for the Garcia−Fair model.

As explained above, the criterion for choosing the best neural network is its ability to generalize, i.e., to make predictions for fresh cases that were not used during the learning phase. Consequently, for a one hidden layer straightforward neural network, the best generalization will be obtained with a 12 hidden nodes network trained with 200 epochs.

The absolute average errors averaged on the 10 training sets is equal to 7.79%, which is a satisfying approximation for the tray efficiency. In fact, the average absolute error of the fundamental model on the training data is about 18%.

Results

Performance of the Neural Network Model for Sieve-Tray Point Efficiency Prediction. To illustrate the ability of the neural network to generalize, the data were divided into training and test sets: the neural network was trained on 80% (223 points) of the data and tested on 20% (56 points). The number of epochs and the number of hidden nodes were 200 and 12, respectively. A parity plot of the neural network model predicted sieve-tray point efficiency vs the experimental measured values for all points in the testing set appears in Figure 4.

The absolute average error is 5.11%. Over the 56 cases tested, there are only four points whose errors were larger than 10% and only one whose error was larger than 25% (27.5%). The performance of the neural network was compared with the performance of the fundamental model developed by Garcia and Fair (Figure 5): the theoretical model was evaluated on the same points as the purely empirical model. The average absolute error is 12.9% on the same test set.

The neural network seems to generalize and make accurate predictions more effectively than the fundamental model. However, such a statement needs to be qualified. Prediction results depend on the sets on which the neural network was trained and tested. For instance, the performance of the 12 hidden node networks trained with 200 epochs (Table 2) yields predictions with varying accuracy.

The semiempirical model has fewer fitting parameters than the neural network model. This may cause the poorer fit exhibited in Figure 5. A "perfect" fundamental description of the underlying physics of the mass-transfer process would yield a perfect fit of the data. Unfortunately, the semiempirical model is not a perfect description and relies, to some degree, on an error minimization to determine model constants.

Dependence of the Weights on Initial Values. The optimization of the weights consists of reducing a target function defined as the distance between the values predicted by the neural network and the experimental values. The target is a function of the weights. Every optimization starts from a random point (random values are given to the weights). The algorithm should provide the optimal point. However, the optimization may stop before the minimum is reached. In fact, it was observed that, to avoid overfitting, the number of iterations has to be limited. Consequently, depending on the "starting point", for the same number of iterations, every "optimization" gives a unique neural network. Moreover, the target function might have a local minimum, and through the optimization process, a "valley" corresponding to a local minimum could be obtained. The performances of 10 different neural networks, trained on the same training set and tested on the same test set (90% and 10% of the data, respectively), are reported in Table 2.

The boldface values correspond to predictions whose error is larger than 10%. They will be considered as "bad" predictions. The performance of neural networks trained on the same data set will depend on the initial weights: some neural networks, for some cases, make "good" predictions (less than 5% error), while some others make "bad" predictions (more than 10% error). Results were obtained by averaging several neural networks. The errors obtained by averaging n neural networks (the n first neural networks whose performances are presented in Table 2) are reported in Table 3.

For some cases, the accuracy of the prediction strongly depends on the neural network used for the prediction. It may happen that an error is larger than 10%, whereas the error is lower than 5% when using other neural networks. Averaging the efficiency provided by several neural networks makes it possible to avoid these "punctual" large errors, particular for some cases (consider cases 1, 3, 5, 6, 9, 11, 12, 14, 15, 18, and 22−24 in Table 2). Moreover, positive and negative errors are balanced. Thus, the performance of a neural network combination is, on average, better than the performance of a single neural network (on average, the error on the test set is 6.23%, which is more than any neural network combination).

Table 2. Performances of Different Neural Networks Trained on the Same Learning Set

		neural network index									
	case	1	2	3	4	5	6	7	8	9	10
relative error on the test cases	1	7.45	5.72	**10.85**	7.46	8.5.1	3.85	6.91	8.68	7.36	8.47
	2	1.37	2.36	1.28	0.81	1.69	0.22	3.68	1.41	0.45	1.11
	3	−7.14	2.73	−1.33	**−13.2**	−7.04	5.63	−8.49	−1.21	**−21.8**	−9.27
	4	−0.73	0.26	−1.19	−1.1	−0.03	0.53	1.6	−0.45	−1.31	−1.13
	5	5.64	6	−0.4	3.79	**11.02**	4.98	3.11	2.23	4.37	3.06
	6	−5.58	−2.96	**−10.7**	−6.64	−3.62	−1.04	−5.61	−6.08	−5.4	−6.49
	7	−5.13	3.96	**−64.4**	**−30.8**	**−30.1**	**−16.6**	**−89.5**	6.64	**−35.2**	**−28.9**
	8	8.96	−4.01	3.98	−1.89	6.66	8.27	−1.2	−5.27	−6.66	4.22
	9	−2.09	**−13.4**	−2.8	2.02	7.75	−6.18	−4.16	−10.8	−12	−1.93
	10	4.09	1.13	1.24	−2.57	1.66	−5.11	−4.47	0.62	−2.42	−3.81
	11	−0.06	−1.72	−2.13	−3.18	−3.24	**−22.5**	**−13**	**22.66**	−7.85	−6.19
	12	−0.54	−9.81	**−10.1**	−3.08	−7.23	−1.88	−1.35	−7.76	**−12.1**	**−10.6**
	13	2	−3.63	−3.67	−8.4	−3.17	7.83	−1.87	2.03	−3.6	5.13
	14	−3.59	5.12	0.06	0.75	0.64	−2.83	0.49	**−11.4**	1.73	0.75
	15	2.1	−0.28	−5.1	−2.46	−1.53	**−11.3**	**−11.4**	**10.38**	−6.1	−1.26
	16	9.56	**−18.6**	**−11.3**	**−20**	−7.88	−6.62	7.02	−0.38	−4.56	−5.72
	17	−1.08	0.93	1.33	−1.07	−0.47	2.28	−1.57	0.46	0.05	−0.06
	18	−4.76	−3.93	**−10.2**	−6.07	−5.8	−4.94	−5.27	−2.88	−4.61	−6.05
	19	1.47	4.84	−2.10	2.41	1.46	1.53	1.24	−1.51	4.37	−5.07
	20	−3.36	−4.33	−8.87	−5.27	−5	−3.72	−7.51	−6.91	−6.63	−7.13
	21	**−17.7**	**−14.3**	0.99	**−22.8**	**−10.9**	**−24**	−1.95	−7.43	**−40**	**−26.8**
	22	−4.26	−4.6	**−10.8**	−6.66	−6.22	−2.67	−5.07	−3.21	−3.63	−6.56
	23	4.79	1.59	1.98	**10.75**	1.28	1.04	1.33	2.37	**11.87**	8.6
	24	2.79	0.05	8.07	**13.06**	−1.04	3.5	1.21	−0.64	−4.36	5
	25	−1.09	0.86	2.07	−1.09	0.14	3.81	−1.51	0.07	−1.54	−0.51
	26	−7.73	−6.82	−8.04	−8.18	−7.18	−7.66	−5.59	−7.59	−8.43	−8.02
	27	2.02	8.4	4.38	6.42	−0.07	3.85	0.58	−6.21	7.16	2.11
	28	**11.09**	6.92	**15.62**	**17.09**	6.78	9.67	6.79	7.33	8.42	**10.21**
absolute average error on the test set		4.58	4.97	7.32	7.46	5.29	6.21	7.27	5.16	8.36	6.58

Table 3. Error Obtained by Averaging *n* Neural Networks

		n =								
	case	2	3	4	5	6	7	8	9	10
relative error on the test cases	1	6.58	8	7.87	8	7.3	7.25	7.43	7.42	7.52
	2	1.86	1.67	1.46	1.5	1.29	1.63	1.6	1.47	1.44
	3	−2.2	−1.91	−4.72	−5.19	−3.38	−4.11	−3.75	−5.76	−6.11
	4	−0.23	−0.55	−0.69	−0.56	−0.38	−0.09	−0.14	−0.27	−0.36
	5	5.82	3.75	3.76	5.21	5.17	4.88	4.55	4.53	4.38
	6	−4.27	−6.4	−6.46	−5.89	−5.08	−5.16	−5.27	−5.29	−5.41
	7	−0.59	**−21.9**	**−24.1**	**−25.3**	**−23.9**	**−33.2**	**−28.3**	**−29**	**−29**
	8	2.47	2.98	1.76	2.74	3.66	2.97	1.94	0.98	1.31
	9	−7.75	−6.1	−4.07	−1.71	−2.45	−2.7	−3.7	−4.62	−4.36
	10	2.61	2.15	0.97	1.11	0.07	−0.57	−0.43	−0.65	−0.96
	11	−0.89	−1.31	−1.77	−2.07	−5.47	−6.54	−2.89	−3.44	−3.72
	12	−5.18	−6.82	−5.88	−6.15	−5.44	−4.85	−5.22	−5.98	−6.44
	13	−0.82	−1.77	−3.43	−3.37	−1.51	−1.56	−1.11	−1.39	−0.74
	14	0.76	0.53	0.59	0.6	0.03	0.09	−1.34	−1	−0.82
	15	0.91	−1.09	−1.43	−1.45	−3.09	−4.28	−2.44	−2.85	−2.69
	16	−4.5	−6.78	**−10.1**	−9.65	−9.15	−6.84	−6.03	−5.87	−5.85
	17	−0.08	0.39	0.03	−0.07	0.32	0.05	0.1	0.09	0.08
	18	−4.34	−6.28	−6.23	−6.14	−5.94	−5.85	−5.48	−5.38	−5.45
	19	3.16	1.38	1.63	1.6	1.59	1.54	1.16	1.52	0.86
	20	−3.85	−5.52	−5.46	−5.37	−5.09	−5.44	−5.62	−5.73	−5.87
	21	**−16**	**−10.3**	**−13.4**	**−12.9**	**−14.8**	**−12.9**	**−12.3**	**−15.3**	**−16.5**
	22	−4.43	−6.56	−6.58	−6.51	−5.87	−5.76	−5.44	−5.24	−5.37
	23	3.19	2.79	4.78	4.08	3.57	3.25	3.14	4.11	4.56
	24	1.42	3.64	5.99	4.58	4.4	3.95	3.37	2.51	2.76
	25	−0.11	0.61	0.19	0.18	0.78	0.46	0.41	0.19	0.12
	26	−7.27	−7.53	−7.69	−7.59	−7.6	−7.31	−7.35	−7.47	−7.52
	27	5.21	4.93	5.3	4.23	4.17	3.65	2.42	2.95	2.86
	28	9	**11.21**	**12.68**	**11.5**	**11.19**	**10.57**	**10.16**	9.97	9.99
absolute average error on the test set		3.77	4.82	5.32	5.19	5.09	5.27	4.75	5.04	5.11

From these results, we can draw the following conclusions:

(i) The performance of the neural network strongly depends on the initial values of the weights.

(ii) Averaging the results given by several neural networks provides, on average, better results.

(iii) Some neural networks, on some points, do not provide an accurate prediction (error larger than 10%), while some other neural networks (same structure and same training time) provide an accurate prediction.

Limitations of the Neural Network Model. There are two types of generalizations: interpolation and extrapolation. Interpolation can often be done reliably and applies to cases that are more or less surrounded

Table 4. Range of Values for Each Input Parameter in the Training Set

input param	minimal value in the training set	maximal value in the training set
training set liquid density (kg/m^3)	373	998.4
vapor density (kg/m^3)	0.06	89.12
liquid viscosity (cP)	0.050	1.586
vapor viscosity (cP)	0.0065	0.0177
liquid diffusivity (m^2/s)	5.5×10^{-10}	2.3×10^{-8}
vapor diffusivity (m^2/s)	2.94×10^{-7}	1.58×10^{-4}
surface tension (dyn/cm)	1.1	71.4
slope of equilibrium curve	0.0055	8.6
hole diameter (mm)	3.175	17
weir height (mm)	19	50.8
HA/AA	0.04824	0.14
liquid rate of flux (m^3/(s·m-weir)	2.22×10^{-4}	2.12×10^{-2}
F_{AA} (m/s)(kg/m^3)$^{0.5}$	0.184	3.759

by nearby training cases; if a test case is close to training cases, the correct output for the test case will be close to the correct outputs of those training cases. Extrapolation is, in general, unreliable. In particular, extrapolation occurs when cases are outside the range of the training data. The neural network was tested on fresh cases whose input parameters were in the range reported in Table 4. Thus, predictions must have been made by interpolation, and the predictions are accurate.

If the parameters are outside the range of value presented in Table 4, the neural network model is doing an extrapolation and the prediction is questionable. On the other hand, if the parameters are inside this range, as will be shown, it is not obvious that the neural network is doing an interpolation and that the prediction is reliable.

Six different neural networks (200 epochs and 12 hidden nodes) trained on all of the data were tested on six different cases. The first two cases were in the training database, and the four other cases were not. For every case, every input parameter is in the range of values presented in the previous table. The input parameters may be divided into three groups describing (i) the physical properties of the system, (ii) the sieve-tray geometry, and (iii) operating conditions (weir load, *F* factor, etc.).

To study realistic cases, for each fresh case, each group of input parameters was the same as the group of the first or second training case. The six cases are described in Table 5.

The tray efficiencies predicted by each neural network are represented in Figure 6. On cases 1 and 2, the different neural networks give the same prediction (±5%). In fact, the neural networks were trained on these points. Consequently, the prediction given by each

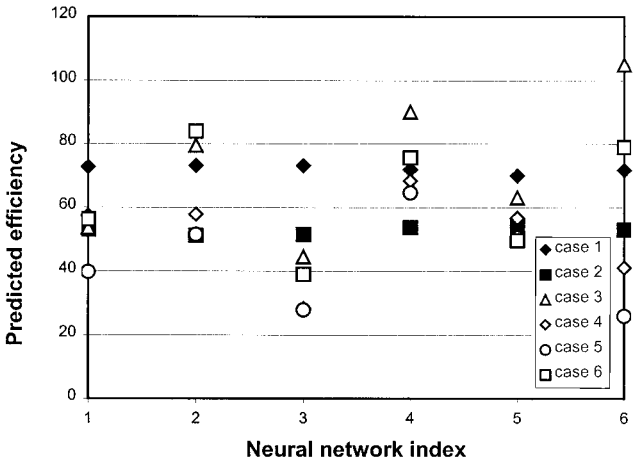

Figure 6. Predictions made by six neural networks on six test cases.

of them is very close to the actual value and, as a result, the standard deviation of the predictions given by the six neural networks on cases 1 and 2 is low (Table 6).

For cases 3−6, we can observe a wide disparity of the predictions made by the different neural networks (see Table 6). On each fresh case, the six neural networks cannot simultaneously make an accurate prediction (less than 20%).

Interpretation of the Results. Generalization is possible as long as the training cases represent a sufficiently representative subset of the set of all cases of interest. The results indicate that, even if a case has all of its parameters inside the range used for training, the database does not necessarily contain cases close enough to allow interpolation. Apparently, the data contain large "holes". When trying to make predictions in a "hole", the neural network is actually doing an extrapolation and the predictions are not accurate. The fact that the input parameters are in the training database range does not guarantee an accurate prediction. In previous work,[2] a sensitivity analysis was performed to identify the key input variables for the neural network prediction of packed column HETP. A similar approach could be utilized for variable reduction for the trayed column efficiency prediction.

Conclusion

A neural network model has been developed for the prediction of the sieve-tray point efficiency. The purely empirical model was tested on data that were not used to train the neural network and yielded very accurate predictions. Therefore, the neural network model is an

Table 5. Six Cases Whose Parameters Are in the Same Range as the Training Cases

	case					
	1	2	3	4	5	6
liquid density (kg/m^3)	386.0	857.0	386.0	857.0	857.0	386.0
vapor density (kg/m^3)	82.56	0.49	82.56	0.49	0.49	82.56
liquid viscosity (cP)	0.050	0.377	0.050	0.377	0.377	0.050
vapor viscosity (cP)	0.0107	0.0080	0.0107	0.0080	0.0080	0.0107
liquid diffusivity (m^2/s)	1.48×10^{-8}	3.21×10^{-9}	1.48×10^{-8}	3.21×10^{-9}	3.21×10^{-9}	1.48×10^{-8}
vapor diffusivity (m^2/s)	2.94×10^{-7}	2.11×10^{-5}	2.94×10^{-7}	2.11×10^{-5}	2.11×10^{-5}	2.94×10^{-7}
surface tension (dyn/cm)	1.1	25.0	1.1	25.0	25.0	1.1
slope of equilibrium	0.9904	0.9143	0.9904	0.9143	0.9143	0.9904
hole diameter (mm)	12.7	12.5	12.5	12.7	12.7	12.5
weir height (mm)	50.8	19	19	50.8	50.8	19
HA/AA	0.0832	0.136	0.136	0.0832	0.0832	0.136
weir load (m^3/(s·m-weir)	8.18×10^{-3}	8.91×10^{-4}	8.91×10^{-4}	8.91×10^{-4}	8.18×10^{-3}	8.18×10^{-3}
F_{AA} (m/s)(kg/m^3)$^{0.5}$	0.385	1.144	1.144	1.144	0.385	0.385

Table 6. Standard Deviation of the Predictions Made by the Six Networks

	case					
	1	2	3	4	5	6
standard deviation	1.21	1.26	22.99	14.35	14.99	18.15

alternative to traditional mechanistic models. However, the good performance of the neural network must be carefully interpreted. The neural network model is not unique: different sets of weights provide a good fit on the experimental data. If all input parameters are in the same range as the parameters in the training database, the accuracy of the prediction is not guaranteed. The prediction is reliable only if the case is similar enough to cases used to train the network. However, considering the results provided by several neural networks (average and disparity of results) made it possible to detect extrapolations and unreliable predictions and to make better predictions than a single neural network. Based on these observations, two MS Excel based programs were developed for the prediction of the sieve-tray point efficiency and structured packing HETP. The structure of the networks and the weights were optimized using the MATLAB software. Then, after optimization, six networks were converted to an MS Excel format. The MS Excel programs represent an improvement of single neural networks and allow the column designer to evaluate the accuracy of the prediction. Moreover, because of their format, the two programs are very easy to use.

Appendix: Development of MS Excel Programs

Motivation and Objectives. According to the results presented above and the results presented in the paper in ref 2, the purely empirical neural network is a very effective modeling tool, offering better predictions than theoretical models. However, the neural network in its current format is not convenient for column designers. The designer needs to be able to run the software (MATLAB format), normalize the input parameters, and train the network before making any predictions. Moreover, even if each parameter is in the same range as the parameters of the database, extrapolation may occur, and in this case, the prediction made by the purely empirical model may be very far from the actual value.

Consequently, two MS Excel programs were developed (one for the sieve-tray efficiency prediction and one for structured packing HETP prediction) which yielded the performance of a neural network, without its disadvantages.

The two programs must meet the following requirements:

(i) They must be based on neural networks whose structure and weights have been previously optimized.

(ii) The input variables do not have to be normalized before prediction.

(iii) They must be universally usable (typically implemented in an MS Excel format).

(iv) The output must warn the column designer when the predictions are not reliable.

Conversion of Neural Network Models to the MS Excel Format. In both cases (the sieve-tray point efficiency and structured packing HETP), the neural networks are two-layer feedforward networks, whose activation function is the hyperbolic function. Such neural networks are entirely defined by two matrixes $\mathbf{A} = a_{ij}[(n + 1) \times m]$ and a column vector $\mathbf{B} = b_j[(m + 1) \times 1]$, where n is the number of inputs plus 1, m is the number of hidden nodes plus 1, a_{ij} is the weight associated with the ith input and the jth hidden node, for $i > 0$, a_{0j} is the bias associated with the jth node, b_j is the weight associated with the jth node and the output node, for $j > 0$, and b_0 is the bias associated with the output node.

Therefore, if X_i is the ith normalized input, the output value of the hidden nodes j, Y_j, is expressed by

$$Y_j = \tanh[a_{0j} + (\sum_{i=1}^{i=n} a_{ij} X_i)]$$

and the final output value, Z, is expressed by

$$Z = b_0 + \sum_{j=1}^{j=m} b_j Y_j$$

Initially, the structure (number of hidden nodes) and the values of weights and biases will be optimized with the specific MATLAB software. Then, the neural network will be trained on all of the data. Finally, the optimized weights and biases will be extracted. Then, because a "frozen" neural network is nothing other than an analytical function involving multiplications, summations, and exponential functions, it will be possible to convert the optimized neural network to an MS Excel format.

Structure of an MS Excel Program Representing a Neural Network. The MS Excel neural network has the following structure: a row vector $\mathbf{x}(n)$, where x_i is the ith nonnormalized input variable, given by the column designer; a row vector $\bar{x}(n)$, where $\bar{x}_i(n)$ is the average on the training set of the ith input variable; a row vector $\boldsymbol{\sigma}(n)$, where σ_i is the standard deviation on the training set of the ith input variable; a row vector $\mathbf{X}(n+1)$, where, for $i > 0$, X_i is the normalized input corresponding to the ith input parameter (nonnormalized) given by the user, with X_i is expressed by

$$X_i = \frac{x_i - \bar{x}_i}{\sigma_i}, \quad \text{for } i = 0,\ X_0 = 1$$

a matrix $\mathbf{A}'(m,n+1)$, transposed of the weight matrix $\mathbf{A}$; a matrix $\mathbf{P}(m,n+1)$, where $p_{ji} = a_{ij}X_i$; a column vector $\mathbf{Q}(m)$, where

$$q_j = \sum_{i=0}^{n} p_{ji}$$

(therefore, q_j is the input value of the hidden node number j); a column vector $\mathbf{Y}(m+1)$, where, for $j > 0$, $Y_j = [\exp(q_j) - \exp(-q_j)]/[\exp(q_j) + \exp(-q_j)]$ and $Y_0 = 1$; a column vector $\mathbf{B}(m+1)$, where, for $j > 1$, b_j is the weight relating the jth hidden node to the output node and b_0 the bias; a column vector $\mathbf{R}(m+1)$, where $r_j = b_j Y_j$, the output value:

$$Z = \sum_{j=0}^{j=m} r_j$$

The row vectors $\bar{x}$ and $\boldsymbol{\sigma}$ are fixed in the program. They were previously calculated from the values of the

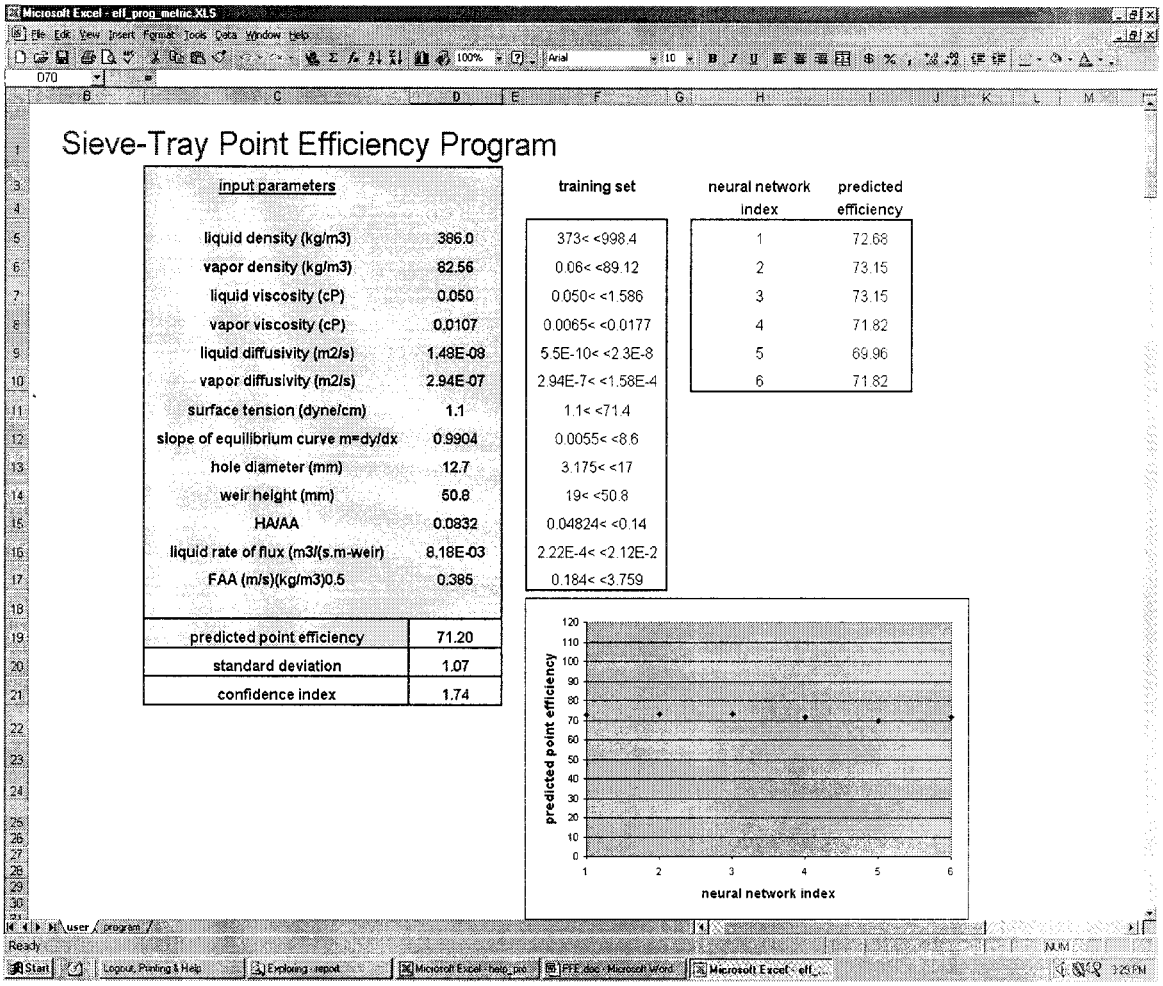

Figure 7. Interface user/program for the program "sieve tray point efficiency".

input parameters in the whole training set. The matrix $\mathbf{A}'$ and the column vector **B** are fixed in the program. They were previously optimized with the MATLAB software. The row vector **x** may be changed by the column designer.

The row vectors $\bar{x}$ and **X**, the matrix **P**, the column vectors **Q**, **Y**, and **R**, as well as the output *Z*, are automatically updated.

Improvement of the Single Neural Network Model. Averaging the results given by several neural networks provides, on average, better results than a single network. Moreover, it makes it possible to avoid "punctual" large errors, occurring for the prediction of some cases using some neural networks. So, in the Excel programs, the output value will be the average of the predictions made by several networks and will be designated the "predicted efficiency".

As shown in Table 5, increasing the number of networks used does not necessary improve the overall performance of the model. Therefore, six neural networks were chosen for inclusion in the Excel program. The number of networks was not optimized, but by using the Excel program "sieve tray point efficiency" with six networks, "punctual" errors were avoided. Moreover, it made it possible to detect unreliable predictions. The same comments may be made about neural network modeling of structured packing HETP, and the program "structured packing HETP" will also consist of six neural networks.

Detection of Unreliable Predictions. If several networks are doing interpolation, i.e., if they are doing predictions for cases similar to the cases used to train the network, the error made by every neural network is, on average, small (less than 10%). Consequently, the standard deviation of the predictions made by several neural networks in such cases is small. When the neural networks are doing unreliable predictions, their predictions might be very different and, as a result, the standard deviation of the different predictions will be large. Therefore, the standard deviation of the prediction made by different neural networks may detect when neural networks are doing extrapolation. If the standard deviation is "small", the prediction is likely to be accurate; if it is "large", the predictions made by some of the neural networks, if not all of them, are misleading. The notions of "small" and "large" will depend on the prediction accuracy expected by the column designer.

A "confidence index" was developed for the predictions. The confidence index is defined by

$$\text{confidence index} = \max\left(\frac{(\max\{Z_i\}_{i=1,\dots,6} - Z)}{Z} \times 100, \frac{(Z - \min\{Z_i\}_{i=1,\dots,6})}{Z} \times 100\right)$$

$\{Z_i\}_{i=1,\dots,6}$ represent the set formed by the predictions made by the six neural networks (Z_1, Z_2, Z_3, Z_4, Z_5, and

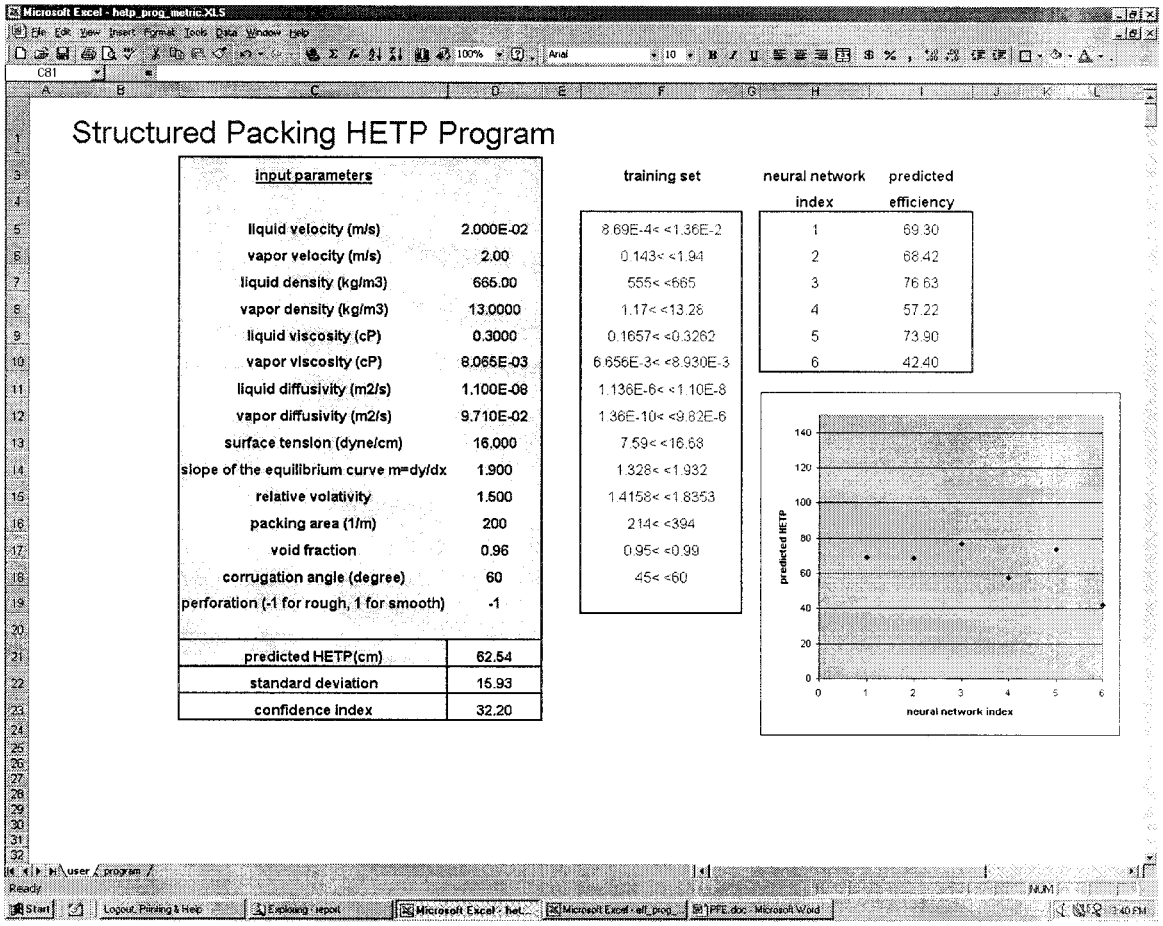

Figure 8. Interface user/program for the program "structured packing HETP".

Z_6), and Z is the average of the predictions made by the six neural networks. max $\{Z_i\}_{i=1,...,6}$ is the largest number in the set $\{Z_i\}_{i=1,...,6}$ and min $\{Z_i\}_{i=1,...,6}$ the smallest number. Assuming that the actual output value is comprised between max $\{Z_i\}_{i=1,...,6}$ and min $\{Z_i\}_{i=1,...,6}$, the confidence index is the most pessimistic estimation of the error made by the model.

Structure of the Final Programs. The two MS Excel programs ("sieve tray point efficiency" and "structured packing HETP") have exactly the same structure. They contain two Excel spreadsheets: "user" and "program". The spreadsheet "user" is the interface between the column designer and the program where the designer gives the program the values of the input parameters. In the spreadsheet "program", six neural networks are implemented according to the structure described above. In the spreadsheet "user" (see Figures 7 and 8), the column designer gives the program the input parameters. For every parameter, the range of values in the training database is indicated. The values given by the column designer are reported to the second spreadsheet "program". Every network calculates the output value. In "user", the predictions made by the six neural networks are reported and represented by a plot. The average predicted efficiency, called "predicted efficiency", the standard deviation, and the confidence index are also reported in "user". When the input parameters are changed, the predictions are automatically updated. The sizes of the MS Excel programs "sieve tray point efficiency" and "structured packing HETP" are 130 and 200 kB, respectively.

List of Symbols

A_A = active (bubbling) area of the tray, m^2
A_F = fractional free area hole area, A_H/A_A
A_H = hole area of the tray, m^2
a_i = interfacial area for mass transfer, m^2
a'_i = interfacial area per volume, 1/m
d_H = hole diameter, m
D_L = liquid diffusivity, m^2/s
D_V = vapor diffusivity, m^2/s
E_{MV} = overall tray efficiency
E_{OC} = overall column efficiency, fractional
E_{OG} = point efficiency (gas concentrations), fractional
F_{AA} = superficial F factor based on active (bubbling) area: $F_{AA} = U_A\rho_V^{1/2}$, $(m/s)\cdot(kg/m^3)^{1/2}$
G_M = mass flow rate of gas, kg/s
h_w = weir height, mm
k_G = gas-phase mass-transfer coefficient, m/s
k_L = liquid-phase mass-transfer coefficient, m/s
L_M = mass flow rate of liquid, kg/s
m = slope of the equilibrium curve dy/dx
N_L = number of liquid-phase mass-transfer units
N_G = number of gas-phase mass-transfer units
N_{OG} = number of overall mass-transfer units
Q_{lw} = volumetric liquid flow rate/weir length or weir load, $m^3/(s\cdot m)$
t_G = mean residence time of gas in dispersion, s
t_L = mean residence time of liquid in dispersion, s
U_A = superficial gas velocity based on the active area, m/s
x = local liquid-phase concentration mole fraction
y = local gas-phase concentration mole fraction
y^* = gas-phase concentration mole fraction in liquid with liquid-phase concentration x

Greek Symbols

α = relative volatility
λ = ratio of slopes, equilibrium to operating line: $\lambda = m/(L_M/G_M)$
μ_L = liquid viscosity, cP
μ_V = vapor viscosity, cP
ρ_L = liquid density, kg/h
ρ_V = vapor density, kg/h
σ = surface tension, N/m

Subscripts

G = gas phase
L = liquid phase
n = tray n
$n - 1$ = tray above tray n

Literature Cited

(1) Garcia, J. A.; Fair, J. R. A Fundamental Model for the Prediction of Distillation Sieve Tray Efficiency. 1. Database Development. *Ind. Eng. Chem. Res.* **2000**, *39*, 1809.

(2) Pollock, G. S.; Eldridge, R. B. Neural Network of Structured Packing Height Equivalent to a Theoretical Plate. *Ind. Eng. Chem. Res.* **2000**, *39*, 1520−1525.

(3) Mehrotra, K.; Mohan, C. K.; Ranka, S. *Elements of Artificial Neural Networks*; MIT Press: Cambridge, MA, 1997.

(4) Lewis, W. K. Plate Efficiency of Bubble Cap Columns. *Ind. Eng. Chem.* **1936**, *28*, 339.

(5) Chan, H.; Fair, J. R. Prediction of Point Efficiencies on Sieve Trays. 1) Binary Systems. *Ind. Eng. Chem. Process Des. Dev.* **1984**, *23*, 814.

(6) Locket, M. J. *Distillation Tray Fundamentals*; Cambridge University Press: Cambridge, England, 1986.

(7) Prado, M.; Fair, J. R. Fundamental Model for the Prediction of Sieve Tray Efficiency. *Ind. Eng. Chem. Res.* **1990**, *29*, 1031.

(8) Garcia, J. A.; Fair, J. R. A Fundamental Model for the Prediction of Distillation Sieve Tray Efficiency. 2. Model Development and Validation. *Ind. Eng. Chem. Res.* **2000**, *39*, 1818.

(9) Rocha, J. A.; Bravo, J. L.; Fair, J. R. Distillation Columns Containing Structured Packings: A Comprehensive Model for their Performance. 1. Hydraulic Models. *Ind. Eng. Chem. Res.* **1993**, *32*, 641−651.

(10) Rocha, J. A.; Bravo, J. L.; Fair, J. R. Distillation Columns Containing Structured Packings: A Comprehensive Model for their Performance. 2. Mass Transfer Models. *Ind. Eng. Chem. Res.* **1996**, *35*, 641−651.

Received for review June 6, 2001
Accepted March 13, 2002

IE010499L