Predictive Neural Network Modeling of pH and Electrical Conductivity in Deep–Trough Hydroponics

K. P. Ferentinos, L. D. Albright

Abstract. *A model is presented that predicts pH and electrical conductivity (EC) changes in the root zone of lettuce (*Lactuca sativa *cv. Vivaldi) grown in a deep–trough hydroponic system. A feedforward neural network is the basis of that modeling. The neural network model has nine inputs (pH, EC, nutrient solution temperature, air temperature, relative humidity, light intensity, plant age, amount of added acid, and amount of added base) and two outputs (pH and EC at the next time step). The most suitable and accurate combination of network architecture and training method was one hidden layer with nine hidden nodes, trained with the quasi–Newton backpropagation algorithm. The model proved capable of predicting pH at the next 20–minute time step within 0.01 pH units and EC within 5 μScm^{-1}. Simpler prediction methods, such as linear extrapolation and the "lazy man" prediction (in which "prediction" is the value of the previous time step), gave comparable accuracy much of the time. However, they performed poorly in situations where the control actions of the system had been activated and produced relatively rapid changes in the predicted parameters. In those cases, the neural network model did not encounter any difficulties predicting the rapid changes. Thus, the developed model successfully identified dynamic processes in the root zone of the hydroponic system and accurately predicted one–step–ahead values of pH and EC.*

Keywords. *Neural network modeling, Hydroponics, pH, Electrical conductivity, Backpropagation algorithms.*

Many research reports describe nutrient uptake by plants and the dynamics of growth (see, for example, Barber and Silberbush, 1984; Wild and Breeze, 1981; Marshall and Porter, 1991; Lawlor, 1991). Models of certain dynamic processes (such as average shoot or root concentration, transpiration, nutrient uptake, etc.) exist, but generally not at the whole–plant level and seldom in a form useful for environmental control or fault detection.

Neural networks (NN) have been used to model a variety of biological and environmental processes (e.g., Bhat et al., 1990; Seginer et al., 1994; Thompson and Kramer, 1994; Sridhar et al., 1996; Lacroix et al., 1997; Lin and Jang, 1998; Altendorf et al., 1999; Hong et al., 2000), but not in the specific area of plant cultivation. Knowledge of the dynamics of interacting biological systems within hydroponically grown plants is limited. Moreover, hydroponic systems can be monitored with a level of detail that permits one to collect extensive sets of data about the system. Thus, the NN approach shows promise as a means to avoid the need to model internal plant processes and still achieve a prediction capability suitable for control and fault–detection algorithms.

Hydroponic systems provide an opportunity to monitor and control the growing process of plants and characterize their interactions with their surrounding microenvironments. Multiple sensors in the root zone make monitoring of nutrient solution characteristics possible. Information about the shoot environment is also available from sensors that monitor environmental conditions inside the greenhouse. This information, in the form of data rather than analytical expressions and dynamical relationships, provides the NN model the ability to optimize its parameters (weights and thresholds) in order to identify and simulate the real process in the best possible way.

The objective of this work was to develop and validate a neural network model able to predict pH and EC values in the nutrient solution of hydroponically grown lettuce. Use of the model will provide a first step toward solving the greater problem of using such information to develop fault–detection methods for greenhouse crops.

Materials and Methods

The growing plant of the modeled system was lettuce (*Lactuca sativa* cv. Vivaldi), and the growing system was deep–trough hydroponics in a greenhouse. Information about the physical process was collected by measuring the most important parameters that affect the dynamics of nutrient solution pH and EC (fig. 1). These parameters can be divided into three categories:

- System variables. These are the measured parameters of the system: pH, electrical conductivity (EC), and the temperature of the nutrient solution.
- Indoor environment disturbances. These are measured environmental parameters that affect the growing plants: air temperature and relative humidity, which are measured above the canopy, and the photosynthetically active radiation (PAR) integral (over time) at plant level.

Article was submitted for review in February 2001; approved for publication by the Information & Electrical Technologies Division of ASAE in August 2002.

The authors are **Konstantinos P. Ferentinos**, **ASAE Member Engineer**, Graduate Assistant, and **Louis D. Albright**, **ASAE Fellow Engineer**, Professor, Department of Biological and Environmental Engineering, Cornell University, Ithaca, New York. **Corresponding author:** Konstantinos P. Ferentinos, 3 Androu Street, Athens 16233, Greece; phone: +30–210–764–8288; e–mail: kpf3@cornell.edu.

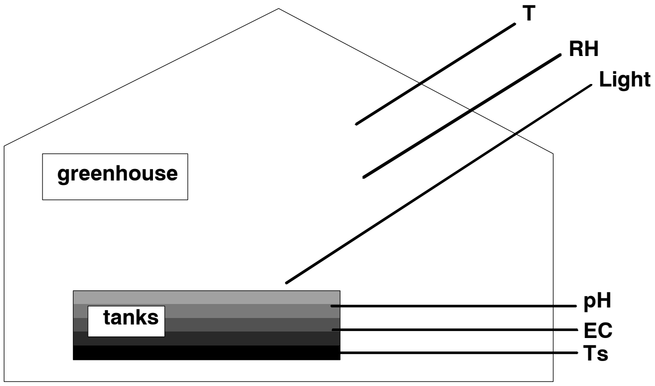

Figure 1. The measured parameters of the system: T = air temperature, RH = relative humidity, Light = light intensity, EC = electrical conductivity, Ts = temperature of nutrient solution.

- Control signals. These are the signals for adding acid or base.

A fourth category, plant age, was also included, as explained later in this article.

Tests were completed in one section of the Kenneth Post Laboratory (KPL) Greenhouses at Cornell University in Ithaca, New York. That greenhouse section has a floor area of approximately 85 m^2 and it is fully equipped with a staged ventilation system, an evaporative cooling system, a supplemental lighting system, and a movable shading system. The deep–trough hydroponic system that was used consisted of three small growing ponds (tanks). Each pond has an area of approximately 1.0 m^2 and a volume of 0.4 m^3 and is completely independent of the others. In this way, the systems can be monitored and controlled in parallel, and three data sets can be obtained simultaneously. Inside each pond are pipes with small, equally spaced holes that serve two important functions:

- To circulate nutrient solution so it remains well oxygenated by dissolved oxygen added through these pipes by a pump and electronically controlled solenoid valves.
- To assure that any added acid or base is well distributed in the nutrient solution so roots are not damaged by direct contact with pure acid or base solution.

Lettuce plants were germinated in growth chambers and transplanted into the ponds after ten days. The growing period in the hydroponic system was 25 days, and the plants were harvested 35 days after seeding. Plants were respaced every 10 days after they were transplanted into the ponds. The initial spacing between plants was 100 mm in a square pattern, and final spacing was approximately 185 mm.

The monitoring and control system consisted of a personal computer (PC) running LabVIEW software (available from National Instruments), a data acquisition board connected to the computer, and several meters, sensors, and actuators connected to the board. The operational features of the greenhouse section that affect the environment are monitored and controlled by the central greenhouse computer. The monitored parameters of the hydroponic system were pH, electrical conductivity (EC), and the temperature of the nutrient solution. A metering pump was used for each pond to control the pH of the nutrient solution by adding acid or base.

Neural Network Training

All parameters in a neural network (weights plus biases) can be denoted by a vector:

$$\underline{w} \in W \subset R^P \tag{1}$$

where p is the total number of parameters of the network (Fine, 1999). Training a neural network involves selecting the "best" set of network parameters ($\underline{w}$) that minimize a training error estimator (Gallant, 1993; Kosko, 1992). The error estimator used here was the sum–squared error (SSE). Preliminary training explored how possible architectures performed when trained with different training algorithms. Next, the best combination of network architecture and training algorithm was selected. Finally, the chosen combination was further tested and trained to obtain the best generalization capability. The training methodology used was the backpropagation training algorithm (Rumelhart et al., 1986). Three minimization algorithms were used: steepest descent, quasi–Newton, and the Levenberg–Marquardt algorithm.

In the backpropagation algorithm, the vector of the network parameters ($\underline{w}$) is updated in each epoch (training step) using:

$$\underline{w}_{k+1} = \underline{w}_k - \alpha_k \cdot M_k \cdot \underline{g}_k \tag{2}$$

where

α = learning rate

g = gradient of the error function

M = approximation of the inverse of the Hessian matrix.

This matrix is positive definite in order to ensure the descent. All quantities are for the kth iteration.

In the steepest descent algorithm, $M_k = I$, where I is the identity matrix, and the search for a minimum takes the opposite direction of the gradient ($-g_k$). That is, the direction of the steepest descent of the error function's "surface." In the quasi–Newton algorithm, the positive definite approximation of the inverse of the Hessian M_k satisfies the relationship:

$$\underline{p}_k = M_k \cdot \underline{q}_k \tag{3}$$

where

$\underline{p}_k = \underline{w}_{k+1} - \underline{w}_k$

$\underline{q}_k = \underline{g}_{k+1} - \underline{g}_k$

In the algorithm used in this work, the approximation was made using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method (Fletcher, 1987; Suykens et al., 1996). In the Levenberg–Marquardt algorithm, the inverse of the Hessian is approximated by:

$$\left[\varepsilon \cdot I + J \cdot J^T\right]^{-1} \tag{4}$$

where

ε = small quantity

I = identity matrix

J = Jacobian matrix $[J_{ij}]$ with $J_{ij} = \dfrac{\partial e_j}{\partial w_i}$, where e_j is the error ($y_j - t_j$) from the target value.

A crucial point in all variations of the backpropagation algorithm is the choice of the learning rate (α), which minimizes the error of the next step (epoch). In the large majority of works in the literature, a fixed–value learning rate is used during the training of the NN. Here, an on–line adjustable learning rate was used, as it seemed to perform much better than the constant one. More specifically, in the steepest descent and quasi–Newton algorithms using the adjustable learning rate, convergence was achieved in two–thirds of the time needed for convergence using the constant learning rate. In the Levenberg–Marquardt algorithm, the improvement was slightly better. In the steepest descent algorithm, the Hessian was used to solve for the "best" learning rate at each iteration. For the other two algorithms, the "best" learning rate was calculated with an approximate line search using a cubic interpolation.

All three training algorithms for both one–hidden–layer (1–HL) and two–hidden–layer (2–HL) networks, as well as the Broyden–Fletcher–Goldfarb–Shanno (BFGS) approximation method and the cubic interpolation used by these algorithms, were written in Matlab (code is listed in Ferentinos, 1999.) Inputs and outputs (target values) were standardized to means of zero and standard deviations of unity. Furthermore, the two most common techniques of controlling the complexity of large neural networks (i.e., validation and regularization) were used.

Network Design and Data Sets

Data sets used to train and then test the neural network were collected between May and July of 1999. In all experiments, pH, EC, and nutrient solution temperature were logged, as well as air temperature, relative humidity, and light intensity of the greenhouse environment. The pH set point was 5.8. The set points for temperature were 24°C during the day (6 a.m. to 10 p.m.) and 19°C during the night. The control zone for relative humidity was from 30% to 70%, and the light integral was controlled to 17 moles m^{-2} d^{-1}, with the algorithm presented in Albright et al. (2000). The EC was not controlled by the computer, but was adjusted manually to keep the nutrient solution between 1150 and 1250 $\mu S\ cm^{-1}$ by adding reverse–osmosis water and solution stocks every two days. All these measurements, which constituted inputs to the NN model, were collected every 10 seconds and values averaged over 20 minutes were logged. In addition, the control signals of the system were used as inputs to the NN. The controlled parameter in our case was the pH of the nutrient solutions, as controlled with metering pumps. The pH control signal (u_{pH}) was continuously variable and was logged as the total amount of acid or base added to the solution during each 20–minute period. Actually, the signal consisted of two signals (u_{pH-A} and u_{pH-B}), depending on whether acid or base was added. Of course, when $u_{pH-A} \neq 0$, then $u_{pH-B} = 0$, and vice versa. Finally, one more input, the plant age estimator, was included. The dynamics of the system change with time because of plant growth, so the model should take into consideration the age of the plants. This leads to an adaptive NN, with the plant age estimator being the adaptation parameter. The plant age was measured in 12–hour intervals, and the growing period of the lettuce was 25 days, so plant age estimator values ranged from 1 to 50.

The outputs of the NN were the predicted future pH and EC values of the nutrient solution averaged over the next 20–minute period (time step). These variables give information on the system's condition and response. Thus, the NN has nine inputs and two outputs (fig. 2), and each input variable accrued 1800 values during each 25–day growing period.

The networks that were tested for the model were 1–HL and 2–HL networks. The activation functions that were used and tested were either logistic or hyperbolic tangent functions. The output nodes were linear. The number of hidden nodes in the 1–HL networks that were tested varied from 4 to 10, while in the 2–HL networks several combinations were tried.

The NN model was trained with experimental data collected during one growing period of the plants. The training set consisted of data from two of the ponds of the hydroponic system. Each column of this set had the values of all network variables (inputs and outputs) of a specific time step. Each time step was 20 minutes, and the growing period was 25 days; thus, the training data set had 1800 columns for each pond, or a total of 3600 columns. Several ratios of training data over validation data were tested to find the one leading to the minimum training errors. The ratio that was most appropriate had 2600 columns of data for training and 1000 for validation. The general training process is shown in figure 2 and had two parts. The first part, the preliminary

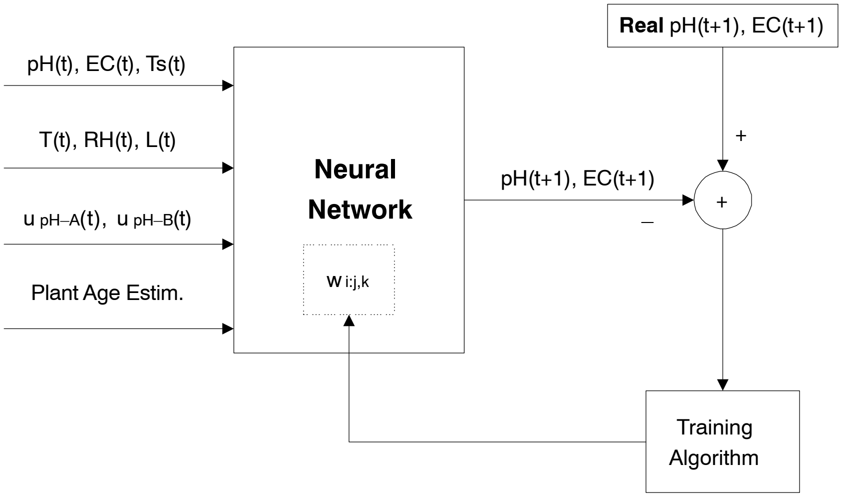

Figure 2. Neural network model inputs and outputs and training process: t = current time, t+1 = next time step, u_{pH-A} = amount of acid added, u_{pH-B} = amount of base added, Plant Age Estim. = plant age estimator factor, $w_{i:j,k}$ = weights and thresholds of neural network.

Table 1. Root mean square (RMS) errors of pH and EC predictions for several 1–HL networks trained with three different backpropagation algorithms: SD = steepest descent, QN = quasi–Newton, LM = Levenberg–Marquardt.

Hidden Nodes	pH Prediction's RMS Errors (pH)			EC Prediction's RMS Errors (μS cm^{-1})		
	SD	QN	LM	SD	QN	LM
4	0.0263	0.0193	0.0300	5.8773	3.5872	8.2644
5	0.0238	0.0150	0.0345	5.4929	3.6924	5.2713
6	0.0245	0.0150	0.0487	4.9394	3.8388	8.9483
7	0.0222	0.0174	0.0461	5.6949	3.5252	6.9018
8	0.0229	0.0150	0.0345	5.4664	3.6011	10.428
9	0.0226	**0.0147**	0.0461	5.4438	**3.4559**	10.325
10	0.0237	0.0159	0.0610	5.5588	3.7276	10.288

training process, investigated which network architecture trained with which algorithm gave the best possible results. This was achieved by training several candidate network topologies (both 1–HL and 2–HL architectures) with all three training algorithms and then comparing the results. The second part of the training process, the basic training process, focused on training the network architecture chosen from the first step with the algorithm that proved best for that specific architecture.

After the training process, the trained NN model was tested in new data (testing data set), and its performance and generalization capabilities were evaluated. Testing data were collected from one pond of the hydroponic system during a separate growing period. The matrix of the testing dataset had 11 rows (9 input and 2 output variables) and 1800 columns; data were collected every 20 minutes over the growing period of 25 days.

Results

Training and Comparison of Networks

In preliminary training, numerous initial conditions and parameters for each network architecture and training algorithm were explored. Two hundred iterations (epochs) were used for these evaluations. From monitoring the error history during these training experiments, it was clear that 200 epochs sufficed to choose the best architecture and training algorithm because the error in each case decreased only slightly after about 180 iterations. Table 1 contains these results for the 1–HL architectures, trained with all three algorithms. The best root mean square (RMS) errors achieved with each algorithm are underlined, for both pH and EC, and the smallest values are **bolded**. Minimum RMS errors for both pH and EC were achieved by a NN with 9 hidden nodes trained with the quasi–Newton algorithm. These minimum RMS errors are 0.0147 for pH and 3.4559 μS cm^{-1} for EC. Thus, from the 1–HL architectures, the best architecture/algorithm combination was a 9 hidden nodes network trained with the quasi–Newton backpropagation algorithm. A comparable procedure for 2–HL network architectures showed that they performed less well. Even the architecture/algorithm combination that appeared best (5 nodes in the first hidden layer and 3 nodes in the second hidden layer trained with the steepest descent backpropagation algorithm) had much larger RMS errors for both pH and EC (0.0241 and 6.2116 μS cm^{-1}, respectively) compared to the 1–HL NN with 9 nodes trained with the quasi–Newton algorithm.

After that preliminary training, the basic training was performed on the selected NN to determine the best possible algorithm parameters and activation functions. Many random initial network parameters were tested. Numerous values of the algorithm parameter λ (coefficient of the penalty term added to the error for regularization and varying by an order of magnitude) were tried. The best results are shown in table 2. All training explorations had a maximum of 300 epochs. Results for both logistic and hyperbolic tangent activation functions (functions of hidden nodes) are presented. From these results it is seen that the value of λ leading to the minimum RMS errors for both pH and EC was 0.01. In addition, the network with a logistic activation function performed better than the network with hyperbolic tangent activation functions.

Table 2. RMSE for pH and EC predictions of the 1–HL neural network with 9 hidden nodes, with several values of the algorithm parameter (λ) and different activation functions, trained with the quasi–Newton algorithm.

	pH RMSE (pH)		EC RMSE (μS cm^{-1})	
λ	Logistic	tanh	Logistic	tanh
0	0.0047	0.0075	3.6004	3.8375
0.001	0.0048	0.0054	3.5163	4.0374
0.01	**0.0046**	0.0058	**3.3750**	3.5708
0.1	0.0067	0.0064	3.5076	3.4582

In summary, the NN model was chosen to have 9 inputs, one hidden layer with 9 hidden nodes (with logistic activation functions), and 2 outputs, and trained with the backpropagation training algorithm using the quasi–Newton multidimensional minimization algorithm, with parameter λ = 0.01.

Testing Results

The testing process consisted of presenting new data to the trained NN model and comparing its output with the real system's outputs. The model gave RMS errors of 0.0072 for the pH and 3.7268 μS cm^{-1} for the EC on the testing data. In figure 3, graphs of the measured pH and EC data and values predicted by the NN model are presented, representing a sample of the testing data (time period of about two days; days 2 and 3 for the pH graph and days 18 and 19 for the EC graph). The days presented are different for the pH and the EC only because these specific days gave clearer and more representative results. The graphs in figure 4 show absolute values of prediction errors for pH and EC. In figure 5, the frequency distribution of absolute prediction errors on the testing data, as well as its cumulative frequency distributions, are presented as another way to show the generalization capabilities of the NN model, specifically:

- Nearly 90% of prediction errors for pH differed from real measurements by less than 0.005, 66% by less than 0.003, and nearly half were between 0.001 and 0.003 pH units.

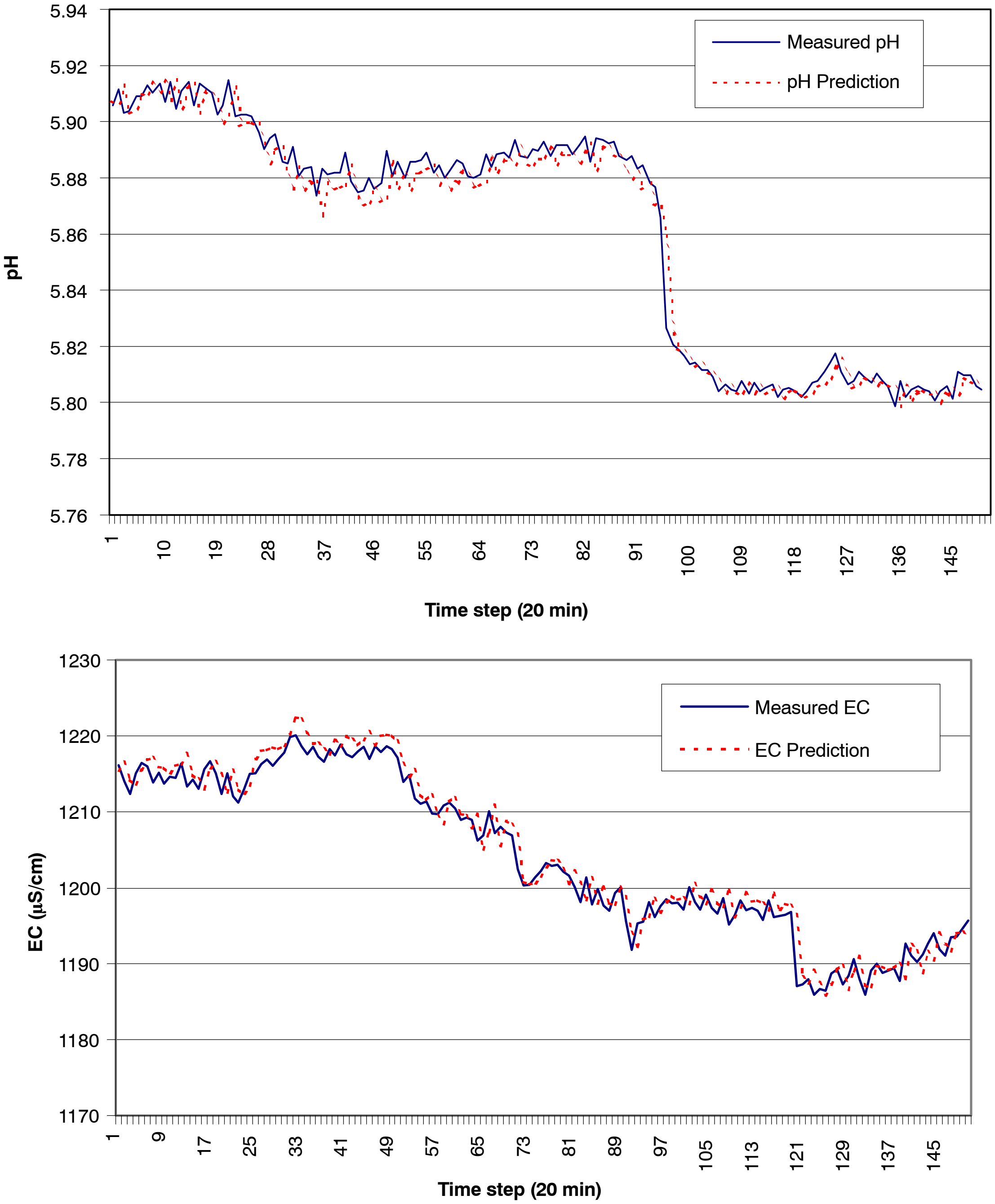

Figure 3. Neural network predictions vs. real measurements on part of the testing data set for pH (days 2 and 3) and EC (days 18 and 19).

- Nearly 82% of prediction errors for EC differed from real measurements by less than 2.0 µS cm^{-1}, and nearly 60% were within 1.0 µS cm^{-1}.

Discussion

The training process of the NN model with data from two ponds collected during a growing period of 25 days seemed to be very successful. The optimal network architecture was obtained after trying several different topologies of one–hidden–layer (1–HL) and two–hidden–layer (2–HL) networks. The simpler architecture that gave the best results on the training data was selected as the final NN model architecture, so that overfitting was avoided and good generalization on unknown data was achieved. This method of choosing the optimal network architecture was materialized using information on the validation error during training. When the validation error was reaching a minimum and starting to increase, the training process was ceased, even though the training error might still be decreasing. In this way, the consequences of overfitting, that is, noisy network outputs with predictions of patterns more complex than the actual ones, were avoided.

For this specific modeling problem, 2–HL network architectures seemed less applicable than 1–HL architectures. This was not unexpected, for 1–HL NN are universal approximators, meaning that they can approximate any nonlinear function. The addition of a second hidden layer in the network can sometimes improve its performance, but not

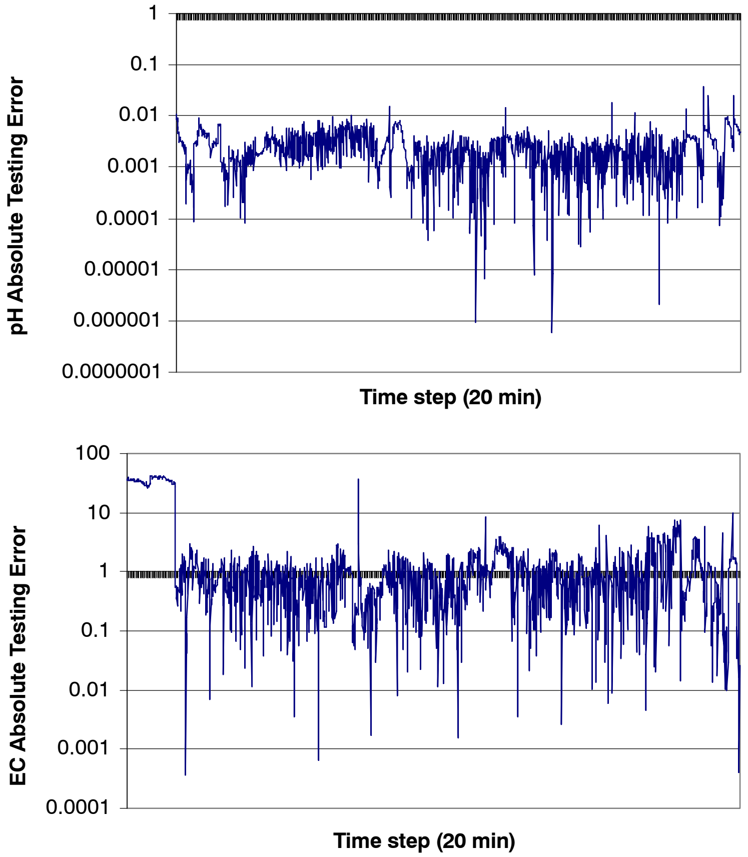

Figure 4. Absolute prediction errors on testing data (25 days) for pH and EC.

necessarily. The preliminary training process showed that the Levenberg–Marquardt (LM) minimization algorithm performed poorly compared to the other two algorithms. The LM algorithm is a powerful algorithm that is much more sophisticated than steepest descent (SD) and quasi–Newton (QN) algorithms. However, even the SD algorithm performed better than the LM in most cases. This proves the general rule that there is no generally accepted "best" minimization algorithm for backpropagation training of neural networks. The performance of the algorithms depends on the specific characteristics of the training data set, such as distributions of values of each variable and specific correlations between variables, as well as correlations between current and future states of each variable. Thus, it appears that the characteristics of this hydroponic system do not particularly favor implementation of the LM algorithm. Another reason that could explain the unexpectedly good performance of the SD algorithm is the use of a learning rate that was continuously adaptive during training. Adaptation was by calculating and using the Hessian, while in the other two algorithms an approximate line search using cubic interpolation was used.

The RMS errors of the NN on the testing data set are similar to those achieved by linear extrapolation prediction or even by a "lazy man" prediction scheme. The linear extrapolation was applied on the testing data set and used a number of past values of pH and EC to extrapolate for the following–step values (predictions). The "lazy man" prediction method simply sets pH(t + 1) = pH(t) and EC(t + 1) = EC(t), i.e., it uses the values of the current time step as "predictions" of the next time step. Even though these methods gave errors similar to the NN errors, the real superiority of the NN model is shown in situations where control parameters influence, to a large degree, the system's response (for example, acid addition that results in rapid pH drop, such as in the pH graph of fig. 3). In these cases, the NN model performed extremely well, in contrast with the linear prediction and the "lazy man" prediction schemes. The latter, as it can easily be imagined, has unacceptable lags in such situations, while the former tends to overshoot and lose the general pattern. These can be observed in figure 6, where a rapid change of pH due to some acid addition is shown. Such rapid changes in the system variables are common in a hydroponic system; therefore, the more sophisticated approach of neural networks is considered justifiable and necessary.

Finally, as it can be noticed by the high absolute testing errors of EC during the beginning of the testing period (fig. 4), the NN model was unable to predict the EC values of the first two days of the testing data. It must be made clear that neural networks can generalize satisfactorily for data similar to the data that they were trained with. The experiment that was used for collecting the testing data was started with lower than normal EC. The NN, for this time of the experiment (plant age estimator from 1 to 4), was trained to give much larger values of EC. However, because EC values of each time step are inputs to the network, its predictions were not as large as the training data values, but also not as small as the actual testing values that it had to predict. This means that the NN has large adaptation and generalization capabilities, but if the testing data are not at all similar to the training ones, then the network cannot have

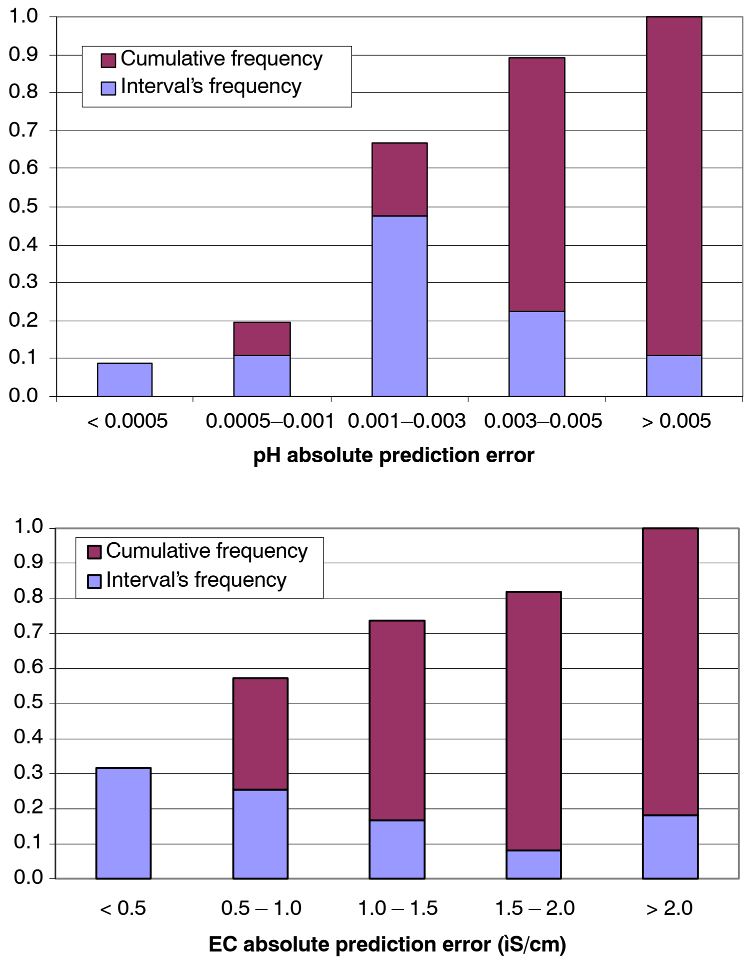

Figure 5. Frequency distributions of absolute prediction errors on testing data for pH and EC.

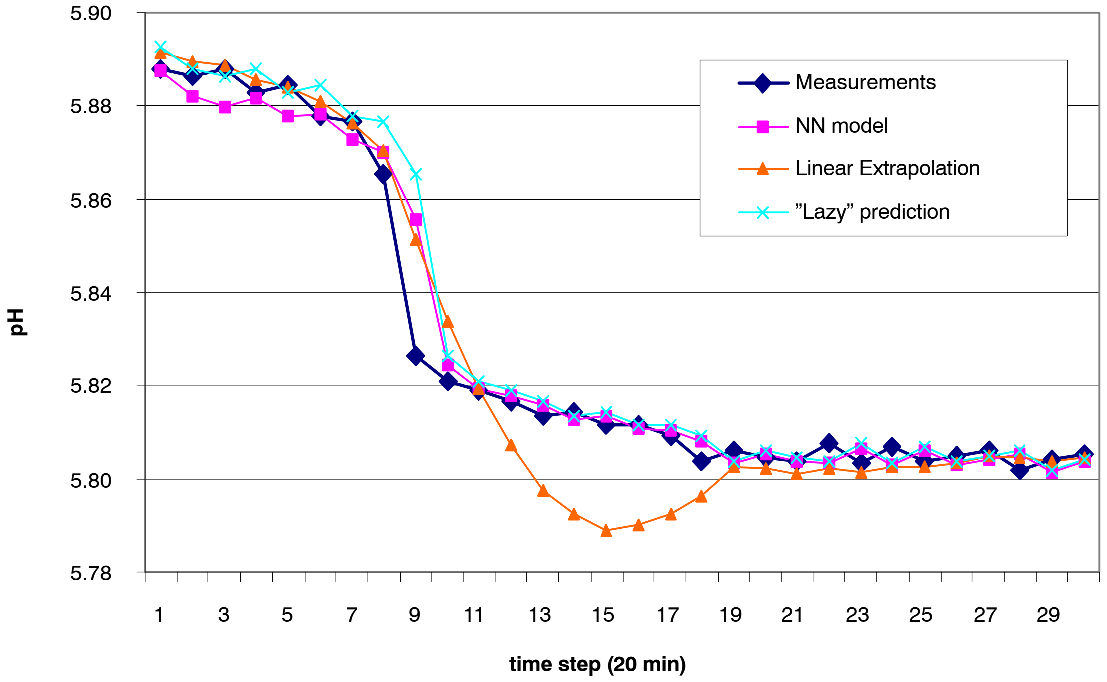

Figure 6. Response of different prediction schemes in relatively rapid pH change.

suitable performance. Once the EC was increased back to normal, the NN model predicted it accurately. It should be mentioned here that the NN model is supposed to function in normal conditions of the hydroponic system. However, it can predict abnormal pH values that arise due to control actions because it has the pH control signals as inputs. Its incapability to predict correctly abnormal EC values is also due to the lack of EC control signals. Thus, it can be concluded that the NN model predicts EC values satisfactorily only when fluctuations of EC fall within the normal range (that is, the range in which the network was trained.) If the NN model must predict abnormal EC values, then the network must be trained with data of abnormal EC behavior.

Conclusions

A new predictive method that uses artificial intelligence to model the pH and EC patterns of the nutrient solution of deep–trough hydroponic systems was developed. More specifically, artificial neural networks were applied successfully in a model that predicts the values of pH and EC of the system. These predictions are considered very important because they can be used by other intelligent systems that ensure the proper and optimal operation of the hydroponic system. In addition, this work builds a link between artificial intelligence and hydroponic systems and with its encouraging results, opens the way to further development and investigation of "intelligent" systems in the field of hydroponics, which will lead to more precise and productive cultivation in hydroponic systems.

From the testing results, it can be concluded that the constructed NN model successfully identified dynamic processes in the root zone of the hydroponic system and accurately predicted values of pH and EC of the nutrient solution at the next time step. More specifically, the following conclusions can be drawn:

- Feedforward neural networks are suitable for modeling pH and EC of hydroponic systems. Once the network is trained with a satisfactory amount of data, it can accurately predict the future state of the system one time step ahead.
- Relatively small network architectures perform better than large ones for this modeling problem. More specifically, 1–HL architectures outperformed 2–HL architectures. The most suitable architecture proved to be that with 9 nodes in one hidden layer.
- The quasi–Newton multidimensional minimization algorithm gives the best training results when used in the backpropagation training algorithm.
- Predicted pH and EC values are sufficiently accurate that the NN model could be considered for use in other systems requiring knowledge of the possible future state of the system. A good example is a fault–detection system. Such a system could read the current state of the process, predict the future state, and compare predictions with the actual process status to detect a possible malfunction. Another example is a control system that "looks" not only at the current status of a process but also into its future state and adjusts control actions according to the predicted future state.
- Simpler prediction methods, such as linear extrapolation or "lazy man" prediction, gave similar RMS errors through the whole testing set, but they perform poorly in situations where the control actions of the system produce relatively rapid changes in the variables to be predicted. Because predictions can be very useful in on–line systems such as the hydroponic system described above, good prediction performance is required. Thus, the NN model would be preferred.

Acknowledgements

This research was supported by BARD Research Project No. IS–2680–96. The authors would also like to thank the Cornell Agricultural Experiment Station for additional funding of this research and Prof. Norman R. Scott of the Department of Biological and Environmental Engineering for his helpful advice.

References

Albright, L. D., A.–J. Both, and A. J. Chiu. 2000. Controlling greenhouse light to a consistent daily integral. *Trans. ASAE* 43(2): 421–431.

Altendorf, C. T., R. L. Elliott, E. W. Stevens, and M. L. Stone. 1999. Development and validation of a neural network model for soil water content prediction with comparison to regression techniques. *Trans. ASAE* 42(3): 691–699.

Barber, S. A., and M. Silberbush. 1984. Plant root morphology and nutrient uptake. In *Roots, Nutrient and Water Influx, and Plant Growth*, 65–87. S. A. Barber and D. R. Boulding, eds. Madison, Wisc.: SSSA, CSSA, ASA.

Bhat, N. V., P. A. Minderman, T. J. McAvoy, and N. S. Wang. 1990. Modeling chemical process systems via neural computation. *IEEE Control Systems Magazine* 24(4): 24–30.

Ferentinos, K. P. 1999. Artificial neural network modeling of pH and electrical conductivity of hydroponic systems. MS thesis. Ithaca, N.Y.: Cornell University.

Fine, T. L. 1999. *Feedforward Neural Network Methodology*. New York, N.Y.: Springer–Verlag.

Fletcher, R. 1987. *Practical Methods of Optimization*. New York, N.Y.: John Wiley and Sons.

Gallant, S. I. 1993. *Neural Network Learning and Expert Systems*. Cambridge, Mass.: MIT Press.

Hong, F., J. Tan, and D. G. McCall. 2000. Application of neural network and time series techniques in wool growth modeling. *Trans. ASAE* 43(1): 139–144.

Kosko, B. 1992. *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. Englewood Cliffs, N.J.: Prentice–Hall.

Lacroix, R., F. Salehi, X. Z. Yang, and K. M. Wade. 1997. Effects of data preprocessing on the performance of artificial neural networks for dairy yield prediction and cow culling classification. *Trans. ASAE* 40(3): 839–846.

Lawlor, D. W. 1991. Concepts of nutrition in relation to cellular processes and environment. In *Plant Growth: Interactions with Nutrition and Environment*, 1–32. J. R. Porter and D. W. Lawlor, eds. Cambridge, U.K.: Cambridge University Press.

Lin, J., and S. Jang. 1998. Nonlinear dynamic artificial network modeling using an information theory based experimental design approach. *Ind. Eng. Chem. Res*. 37: 3640–3651.

Marshall, B., and J. R. Porter. 1991. Concepts of nutritional and environmental interactions determining plant productivity. In *Plant Growth: Interactions with Nutrition and Environment*, 99–124. J. R. Porter and D. W. Lawlor, eds. Cambridge, U.K.: Cambridge University Press.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1986. Learning representations by back–propagating errors. *Nature* 323: 533–536.

Seginer, I., T. Boulard, and B. J. Bailey. 1994. Neural network models of the greenhouse climate. *J. Agric. Eng. Research* 59: 203–216.

Sridhar, D. V., R. C. Seagrave, and E. B. Bartlett. 1996. Process modeling using stack neural networks. *AIChE J*. 42(9): 2529–2539.

Suykens, J., J. Vandewalle, and B. De Moor. 1996. *Artificial Neural Networks for Modelling and Control of Non–Linear Systems*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Thompson, M. L., and M. A. Kramer. 1994. Modeling chemical processes using prior knowledge and neural networks. *AIChE J*. 40(8): 1328–1340.

Wild, A., and V. G. Breeze. 1981. Nutrient uptake in relation to growth. In *Physiological Processes Limiting Plant Productivity*, 331–344. C. B. Johnson, ed. Boston, Mass.: Butterworths.