Using Neural Networks
for Day-Ahead Forecasting

By J. Stuart McMenamin, Ph.D. and Frank A. Monforte, Ph.D.

1. Introduction

Electricity, by its nature, must be produced at the same time that it is consumed. As a result, electric utilities and system operators face a set of very short-term forecasting problems, involving loads in the next hour, the next day, and the next week. With the advent of retail competition, retailers face the problem of forecasting the loads of their customers in each geographic area. And all players in the market face a magnified need for hour-by-hour price forecasts.

In addition to the standard tools of econometrics and time-series analysis, approaches using artificial neural networks (ANN) are being applied to these forecasting problems. ANN-based forecasting systems are being adopted and implemented by utilities, system operators, and retailers across the country. In the case of very short-term problems (hour ahead) these models operate like nonlinear time-series models. For short-term problems (day ahead) these models place much less reliance on autoregressive terms and operate more like nonlinear regression models or transfer-function models.

This paper discusses the nature of these short-term forecasting problems and identifies reasons why the neural network approach is well suited to these applications. It provides a direct comparison of neural network specification with regression approaches. This discussion casts neural networks in terms of conventional statistical concepts, providing a bridge through which direct comparisons can be made.

For each approach, model estimates and forecasts are developed using hourly load data for a Midwestern utility. In the context of regression approaches, several specification issues are addressed, focusing on the identification of important nonlinearities and variable interactions. In the context of the neural network models, the tradeoff between modeling flexibility and forecast power is examined. It is concluded that the BIC statistic provides a good guide for determining the optimal level of model complexity.

In comparing alternative approaches, forecasting power is quantified using out-of-sample MAD and MAPE statistics. It is concluded that all modeling approaches work well for these problems when properly applied. However, the neural network models provide better forecasting power than regression approaches, even when significant effort is taken to structure appropriate nonlinear terms and interactions in regression models.

2. Artificial Neural Networks from an Econometrician's Point of View

A large amount of confusion seems to surround the topic of neural networks. In part, this reflects the fact that a different language is used for neural networks than is used in the more familiar (to forecasters) area of econometrics. In order to bridge this gap, this section focuses on specific elements of the neural network language. The discussion is from the perspective of forecasters trained as econometricians, and the goal is to make the relationship between concepts in neural network modeling and traditional econometric concepts as clear as possible.

Artificial neural networks, as they are used in forecasting, are flexible nonlinear models that can approximate a wide range of data generating process. In general form, for a singlevariable forecasting problem, an artificial neural network looks like this.

$$Y = F(X, B) + u$$

Of course, the X's might be lagged Y's. And there could be several X's. And there could be lots of B's. In this general form, this is nothing new. However, most functions of this general form, including all functions that are normally used in forecasting, do not qualify as neural networks, except as degenerative cases.

Although neural networks can take many forms, the most frequently used form is very specific and can be written as follows:

$$Y^t = B_0 + \sum_{n=1}^{N} \left(B_n \times \frac{1}{1 + e^{-\left(a_{n,0} + \sum_{k=1}^{K} a_{n,k} X_k^t\right)}} \right) + u^t \qquad (1)$$

The thing that makes this different is the repetitive nature of the specification. That is, within the summation, the function in parentheses is repeated N times with exactly the same algebraic form.

In network jargon, equation (1) is called a single-output feedforward artificial neural network, with a single hidden layer, with N nodes in the hidden layer, with logistic activation functions in the hidden layer, and with a linear activation function at the output layer.

To the forecaster, however, it is best to think about this specification as a flexible form that is nonlinear in the variables and the parameters. The form is flexible in that it allows for a wide variety of nonlinearities and interactions among the explanatory variables, and the repetitive specification is the cornerstone of this claim to flexibility.

It is easiest to understand this nonlinear function by looking at a simple example. If the number of explanatory variables (K) is 3, and the number of nodes (N) is 2, then the network function takes the following form.Given values for the explanatory variables and a set of parameter values (the B's, a's, and b's) one can easily compute the predicted values and residuals for each observation. The estimation problem, then, is to find parameters that make the residuals as small as possible.

$$\begin{aligned} Y^t = B_0 + B_1 \times \frac{1}{1 + e^{-\left(a_0 + a_1 X_1^t + a_2 X_2^t + a_3 X_3^t\right)}} \\ + B_2 \times \frac{1}{1 + e^{-\left(b_0 + b_1 X_1^t + b_2 X_2^t + b_3 X_3^t\right)}} + u^t \end{aligned} \qquad (2)$$

Given values for the explanatory variables and a set of parameter values (the B's, a's, and b's) one can easily compute the predicted values and residuals for each observation. The estimation problem, then, is to find parameters that make the residuals as small as possible.

Although function (2) is clearly nonlinear in the X variables and most of the parameters, it has linear components. This can be seen by rewriting the network function as follows:

$$Y^t = B_0 + B_1 \times H_1^t + B_2 \times H_2^t + u^t \qquad (3)$$

In network terms, each H represents a node in the hidden layer. And, for the particular specification used here, the output function, which is shown in (3) is linear in these values. The output function can be specified to be nonlinear in the H's, but for most forecasting problems with continuous dependent variables; there will be no advantage to this additional nonlinearity.

A second linear component is found in the denominator of the H functions. Specifically, the exponent is a linear weighted sum of the input variables, and this can be written as follows:

$$Z_1^t = a_0 + a_1X_1^t + a_2X_2^t + a_3X_3^t \qquad (4)$$

Then, with a bit or rearrangement, gives:

$$H_1^t = \frac{1}{1+e^{-Z_1^t}} = \frac{e^{Z_1^t}}{1+e^{Z_1^t}} \qquad (5)$$

If you are familiar with discrete choice models, you will recognize this as a binary logit, the workhorse of market-share modeling. The linear weighted sum (Z) is the power on e. If Z is a large negative number, H is close to zero. If Z is 0, H is .5. And if Z is a large positive number, H is close to 1.0. In between, it traces out an S-shaped function. A plot of H as a function of Z is as depicted in Figure 1.

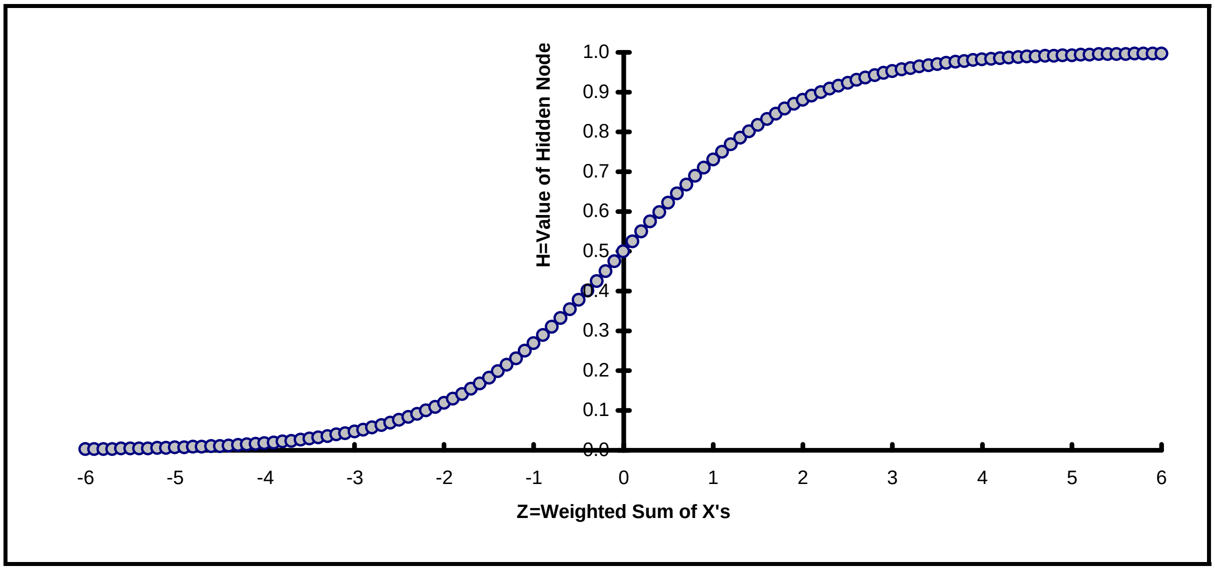

Figure 1: Binary Logistic Function

This also means that there is an S-shaped relationship between each node value (H) and each explanatory variable (X) in that node. This S curve may be positively or negatively sloped, depending on the sign of the slope coefficient on the X variable ($a_{n,k}$ in equation 1).

Further, the specification is automatically interactive. This can be seen by re-writing the exponential as:

$$e^{a_0+a_1X_1^t+a_2X_2^t+a_3X_3^t} = e^{a_0}e^{a_1X_1^t}e^{a_2X_2^t}e^{a_3X_3^t} \qquad (6)$$

As a result, each X variable interacts with all other X’s that do not have zero slopes in the node. This is strength of the specification if the underlying process has multiplicative interactions.

It is true that each X variable appears several times and, in the case of equation (1), in exactly the same algebraic form. At first glance, econometricians will not be comfortable with this idea. It looks like an extreme form of multicollinearity. In network language, the repetitive specification is called parallelism or massive parallelism, and it is one of the strengths of the approach. As you might expect, it raises some serious issues for parameter estimation.

But, if you think about it, we have all put X and X squared on the right-hand side of an equation. So suppose that in the first node, variations in X cause movement in the linear part of the logistic equation (Z between -1 and +1) and in the second node, variations in X are operating in the bottom part of the S shape (Z between -4 and -1). This could happen, depending on the values of the other variables and the parameters involved.

With several nodes in the hidden layer, the specification allows for a variety of nonlinearities and for a range of variable interactions. For example, two logistic curves, one positively sloped and one negatively sloped, can be combined to give a U-shaped response over the relevant range of an X variable. Given this flexibility, the estimation problem is to find a set of specific nonlinearities and specific interactions that are useful for explaining history and for forecasting. The estimation problem is discussed below.

In neural network terms, equation (1) has the following properties.

- It is a feedforward neural net with a single output
- It has one hidden layer with one or more nodes
- It uses logistic (e-based) activation functions in each hidden layer node
- It uses a linear activation function at the output layer.

These terms are discussed further below.

Feedforward Neural Network
Shown in Figure 2 is a depiction of the neural network described by function (2). As shown in Figure 2, the explanatory variables (X) enter at the bottom in the input layer. The logit tranforms appear in the hidden layer. And the result (Y) appears in the output layer.

The idea is that the inputs feed into the nodes in the hidden layer, and there is no feedback. Further, the nodes do not feed sideways into each other. Instead, they feed onward to the output layer. And there is no feedback, delayed or otherwise, from the output layer to the hidden nodes. The absence of feedbacks or node-level interactions makes it a feedforward system.

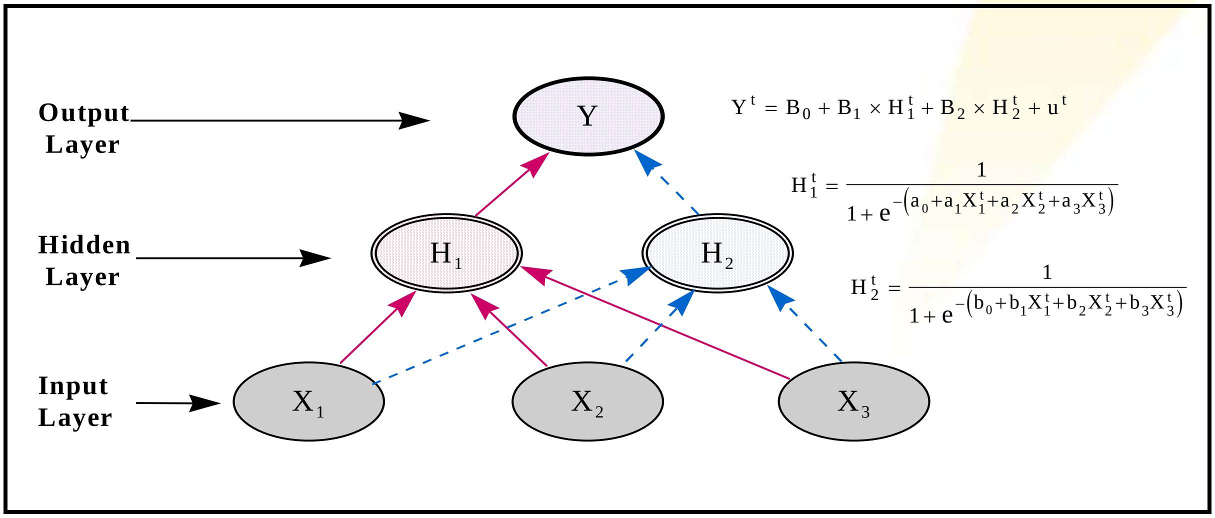
Output Layer
Hidden Layer
Input Layer
Y
H_1
H_2
X_1
X_2
X_3
$Y^t = B_0 + B_1 \times H_1^t + B_2 \times H_2^t + u^t$
$H_1^t = \frac{1}{1 + e^{-\left(a_0 + a_1 X_1^t + a_2 X_2^t + a_3 X_3^t\right)}}$
$H_2^t = \frac{1}{1 + e^{-\left(b_0 + b_1 X_1^t + b_2 X_2^t + b_3 X_3^t\right)}}$

Figure 2: Network Diagram with 3 X's, 2 Nodes

Hidden Layer

Why are the terms in the middle called the hidden layer? There is an answer to this in the neural sciences (see Rummelhart, Hinton, and Williams, 1986). But the answer is of no interest to forecasters. If you look at equation (1), it is clear that nothing is hidden at all. Represented in diagram form, as in Figure 2, there are specific algebraic transformations in the middle layer, but they are hidden only if we decide to hide them somehow. The important and powerful part of the specification is that it is a flexible form that is capable of approximating a wide range of functions. Analogies to the learning process of the brain do not increase or decrease the power of this approximation. Neither do these analogies help us to understand how these equations work in a forecasting context.

Activation Function

Activation function refers to the S-shaped nature of the function in each node and the fact that the boundary values (0 and 1) can be related to on and off. Again, the history of the term is rooted in neural sciences, and it has to do with the requirement that a signal must reach a certain level before a neuron fires to the next level. But it is not a bad description in the forecasting application. For most forecasting problems, if you allow flexibility (more than two nodes), some of the nodes will end up specializing and will activate (take on a value close to 1.0) under specific conditions and take on a value of close to zero otherwise. To see this for a specific problem, all you need to do is plot out the contribution of each node to the total predicted value, and this pattern will become clear.

For forecasting purposes, the hidden layer functions do not need to be logistic functions. Any other S-shaped function could be used, such as an arc tangent or a cumulative normal, and the model behavior would not be changed significantly. The important thing is to use a smooth differentiable function that will be easy to work with for estimation purposes.

Taking this further, it is not necessary to use S-shaped curves at all. For example, bell shaped functions (like the derivative of a logistic curve) could be used in some nodes rather than Sshaped functions, and this would be quite useful in many forecasting applications. However, as you move away from S-shaped curves, the term "activation" becomes less descriptive of functional performance, since the alternative functions would no longer range between zero at one extreme and one at the other. As a result, the hidden node activation functions are sometimes called neuron transfer functions (see, for example, Azoff, 1994, pp 51-55).

Finally, additional nonlinearities could be introduced at the output layer. In fact, for dependent variables that have discrete outcomes, such as a binary (zero/one) variable, a logistic activation function at the output layer would probably be desirable. But for most problems with continuous outcomes, there is no real gain from a further nonlinearity at this level.

3. Estimation Approaches for Neural Networks

In the neural network literature, the process of parameter estimation is called training. The goal of the training process is to find network parameters that make the model errors small. The estimation process is more complicated than for a regression model because the model is nonlinear and because the objective function is relatively complicated.

Many neural network programs use some variant of a method called backpropogation, following the lead of Rummelhart et al. (1986). For the type of forecasting problem addressed here, this approach is unnecessarily slow and cumbersome (see the discussion in Masters, 1995). In what follows, we use a conventional nonlinear least squares algorithm to find parameter solutions. Specifically, we use the Levenberg-Marquat (LM) algorithm in the IMSL library (IMSL, 1994).

The estimation algorithm works as follows. First, a set of random values are assigned to the model parameters. Given this set of parameter values, predicted values and residuals are computed for each observation, along with the derivatives of the residuals with respect to changes in each parameter. The LM algorithm uses this information to change the parameters to new values that will reduce the sum of squared errors. These revised values provide a new starting point, and the revision process is repeated. For the data examined here, the parameter values usually converge within reasonable tolerance within 100 iterations through the data.

Multiple Optima

Parameter estimation is a well behaved process for most common statistical problems, such as finding the solution to a nonlinear regression model or solving the parameters of an ARMA model. However, because of the parallelism property, which reflects the inclusion of multiple nodes in the hidden layer, it can be shown that the least squares objective function for a neural network is extremely complex with a huge number of local optima, as opposed to a single global optimum.

The magnitude of this problem was quantified in a recent paper [Goffe, Ferrier, and Rogers, 1994]. In this paper, the authors examined several estimation problems, including one that involved a neural network like the one specified above. In addressing this problem, the authors became interested in the shape and properties of the objective function. The problem was a modest one, involving a 5-node network with a total of 35 parameters. As alternative solutions were examined, it became apparent that the objective function had a large number of local optima. By pushing out from each optimum point and seeing if estimation returned back to this point, they were able to quantify the size of the region dominated by each optimum point. Taking this region as a fraction of the total parameter space for their problem, the determined that there were about 1019 such points for their problem.

This is a lot of points. As an analogy, consider a two parameter problem. Think of the objective function as looking like the bottom half of an egg carton, with 1 dozen local minimum values of varying depths in each carton. Now think of a football field covered with such cartons. Now think of the state of Texas covered with such cartons. Now think of the surface of the earth covered with such cartons. Now think of 64 times the surface of the earth covered with such cartons. You now have a big enough surface to contain 1019 local optimum points.

So suppose that you pick a random starting point and it turns out to be on the 50 yard line of a high school football field in the middle of Texas. Look around at the football field, out toward the state boundaries, and imagine the 64-earth surface from this point and you start to understand the magnitude of the estimation problem. Somewhere on that surface are several mathematically equivalent global optimum points.

Because of this complexity, it is necessary to explore a wide region of the parameter space to find a relatively good solution. Merely going downhill from a single random starting point to the nearest local optimum will not do the job. This is equally true for estimation approaches based on backpropogation as it is for approaches based on mathematical optimization. Using a multiple-seed approach, we have found that estimation from 20 alternative random starting points is fairly certain to produce several strong solutions for problems of the type studied here.

The rule that we use to select the final model parameters is based on an average of in-sample and out-of-sample error statistics. To develop these statistics, estimation from each random starting point is based on a subset of the sample data. Once a solution is found, these parameters are used to test the power of the estimated parameters, based on the observations that have been withheld from the estimation process. Usually, solutions that perform well in sample also perform well out of sample, but this is not always the case. Especially with a large number of nodes, some of the solutions will be more specialized to the specific cases in the sample, and some will be more stable and more useful out of sample.

The fact that there are many solutions that have comparable performance is neither surprising nor disturbing. This implies that there are a large number nonlinear specifications that provide similar performance. The same result holds for most regression models, in that minor variations in the specification usually do not alter the model results significantly.

Continued Learning

As new data become available, it is natural to re-estimate parameters with the extended sample period. In the neural network literature the update process to incorporate new data is called learning. This terminology stems from the backpropogation method, in which learning is treated as an extension of the process used in training. Although learning is not the focus of this paper, this is a point of confusion for many analysts who believe that learning is a unique property of neural networks.

With linear least squares models, updating the parameters with additional data is a full reestimation of the model, and the new data typically are given the same weighting as the earlier data. For nonlinear least squares, the re-estimation process can begin with the same set of initial guesses that were used to start the original estimation process or it can begin with the solution from that estimation. Either way, for problems with a well-behaved objective function, the final solution will be the single set of parameters that correspond to the global optimum based on the expanded data set.

For neural networks, which are known to have a large number of local optima, the situation is a bit different. In this case, it seems natural to start with the parameters from the training process and re-optimize with the new data included. However, starting at the training solution implies starting with the specific set of nonlinearities and variable interactions represented by that solution. From this starting point, re-estimation with the expanded data typically leads to minor changes in the estimated parameters. This implies that you are staying at the same local optimum at which you started, and that the location of this solution does not move much because of the new data. In this sense, the new data play less of a role than the earlier data. In essence, the functional form was determined in the training process, which looked at many solutions and selected one, and the new data are used only to refine parameters given that functional form. To give the new data equal play with the earlier data, it would be necessary to repeat the entire training process with the expanded data set.

4. Day-Ahead Electricity Demand Forecasting

The dependent variable data is the hourly system load for an electric utility in the Midwest U.S. for 1995 and 1996, giving a total of 17,544 observations. Corresponding data for daily maximum and minimum temperatures, precipitation levels, and wind speed were obtained from the participating utility. In addition to these data, the calendar for that year gives a variety of day-type variables, such as day of the week and the timing of holidays. Finally, data on the time of sunrise and sunset provides other important seasonal information.

A set of 25 neural network models is developed for these data: a daily energy model and 24 separate models for each hour in the day. The predicted value from the daily energy model is used as a right hand side driver in the hourly models. In general, we have found that daily energy is a much smoother series than the individual hourly loads, and hence is easier to forecast accurately. The hourly models are then used to shape the forecasted daily energy.

For purposes of presentation, we focus on the daily energy data. Figure 3 shows a scatter plot of daily load against the average dry-bulb temperature. The points are coded with symbols that separate weekends and holidays from the weekdays in each season. As is clear from this plot, there is significant variation in daily energy for a given temperature, leaving much to be explained by other conditions and calendar variables.

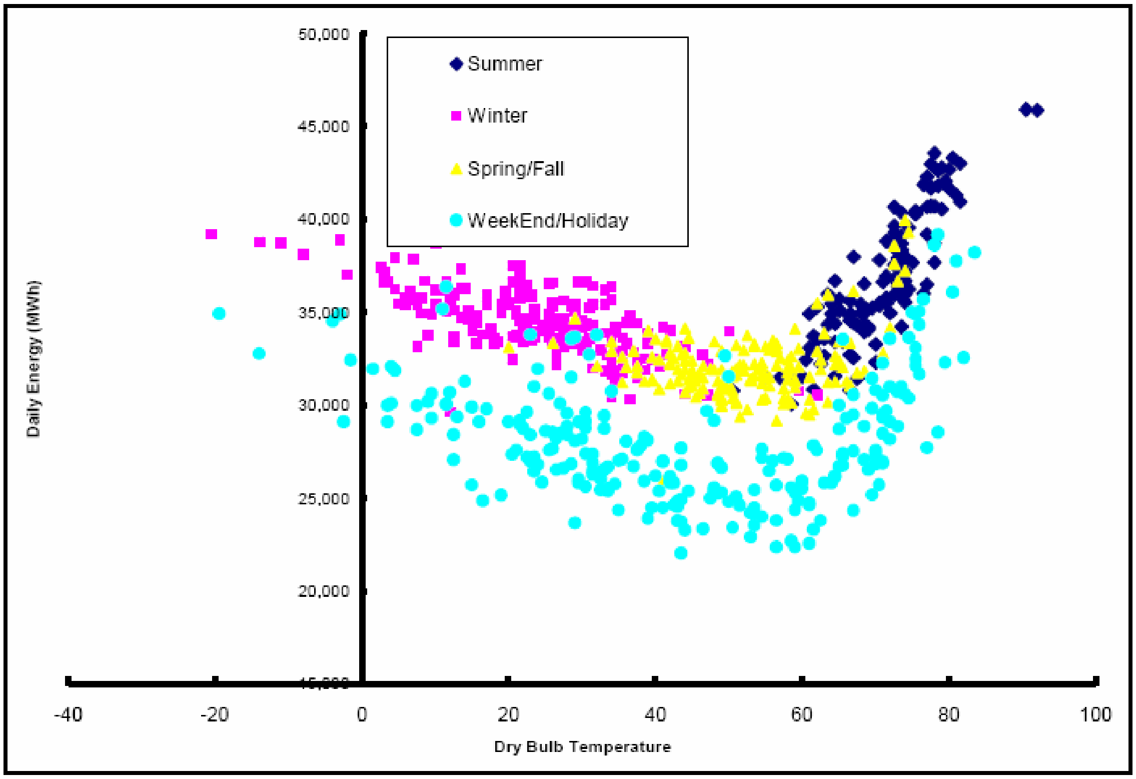

Figure 3: Daily Energy Versus Average Dry Bulb Temperature

Nonlinearities and Interactions

As is evident from Figure 3, the relationship between daily energy and temperature appears to be nonlinear. Specifically, in winter months when it is cold, increases in the temperature appear to reduce the load. This probably reflects reductions in electric heating loads. In contrast, in summer, increased temperature values appear to be strongly correlated with increased loads. Although this slope appears to be positive in spring and fall months, the weather response slope does not appear to be as large as in the summer months.

As indicated in Table 1, there are a significant number of potential explanatory variables. The exact set that is available for modeling depends on the timing of the forecast. For example, for a day-ahead forecast that must be developed by 4 p.m. each day, the most recent value that can be included as a lagged load is the system load as of 3 p.m. Of course, earlier values, such as the load in the morning, can also be included since these values will be known at the time of the forecast.

The difficult part of model specification for an econometric approach to this problem concerns the variable interactions. The most important examples are the interactions between weather variables and calendar variables, the interactions between lagged loads and calendar variables, and the interactions among weather variables. Brief examples of these interactions are discussed below.

- **Weather and Calendar Variables**. It is clear from inspection of the data that an extra degree of temperature has a different impact on a weekday than on a weekend day or holiday. For that matter, the Saturday slope may be different than the Sunday slope, and, as discussed above, the weekday slopes in winter for a given temperature are different than the weekday slopes in summer. These facts suggest that temperature data must be interacted with day-of-the-week variables and seasons, at a minimum.

- **Lagged Loads and Calendar Variables.** Lagged loads are powerful explanatory variables in next-day forecasting exercises. And it is fair to include these variables in a model, since the data values are known at the time of the forecast. However, the relationship between yesterday’s load and today’s load differs significantly across days. On a Monday, the lagged load is for a Sunday, and therefore the slope is different than on a Tuesday, when the lagged load is for a Monday. For some Tuesdays, however, the lagged load is for a Monday holiday. These and other interactions must be allowed in the model so that the differential influence of lagged loads can be estimated across different day types.

- **Weather Variable Interactions.** Laws of thermodynamics and the presence of heating and cooling equipment suggest some important interactions. For example, the influence of increased wind speed in the summer should be negative, due to a lowered cooling loads. However in the winter, increased wind implies wind chill which raises heating loads. These interactions imply that the slopes of weather variables can switch signs depending on other conditions.

Weather	Calendar	Lagged Loads
Daily Max Temperature	Day of the Week	Morning (day-1)
Daily Min Temperature	Month/Season	Afternoon (day-1)
Cumulative Temperature	Holidays	Same hour (day -1)
Temperature Gradient	Days near holidays	Same hour (day-2)
Precipitation	Sunrise & Sunset	
Windspeed		
← Interactions →		

Table 1: Summary of Explanatory Variables

Regression Results
As a point of reference, a linear regression model was estimated with various combinations of the variables and appropriate interactions. The regression model includes all of the factors listed above as explanatory variables. Specifics are as follows:

- Daily high and low temperature variables are included in the model, since this resulted in a better fit than use of the coincident temperature alone.
- For the high and low temperature variables, linear splines are included to allow for nonlinear responses, and different temperature slopes are estimated for week days versus weekend days and for each season.

The results of the regression are depicted in Figure 4. To test the forecasting power of the model specifications, seven days out of every month were set aside for out-of-sample testing. The in-sample mean absolute percent error (MAPE) was 2.66%, which is a good result for this type of data. The out-of-sample MAPE was 2.81%.

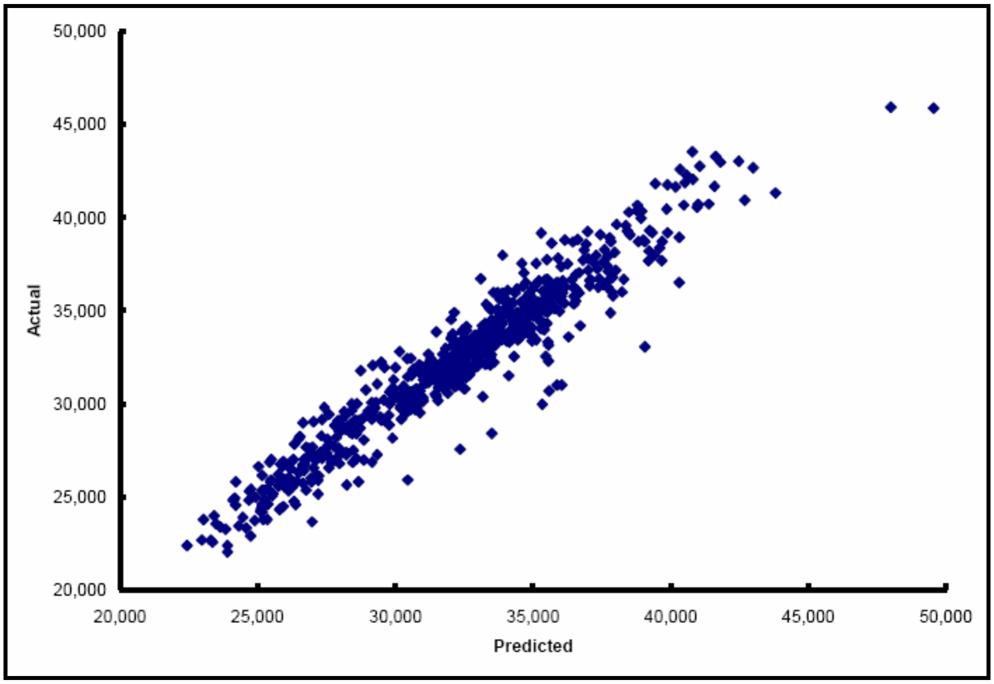

Figure 4: Actual Vs. Predicted Daily Loads – Regression Model

Neural Network Results

Estimation was repeated using the neural network framework described above. Variables included in the model were daily high and low temperatures, precipitation, wind speed, and calendar variables including day of the week, holidays and season. The day-of-the-week and holiday binary variables were included on a linear node, and the weather plus weekend and season binaries were included on two Sshaped activation functions, based on the logistic curve, as shown in equations (1) and (2).

The results of the regression are depicted in Figure 4. The in-sample MAPE was 1.49% and the out-sample MAPE was 1.47%. Both measures are very good for these type of data. The in-sample and out-of-sample fit for the three-node neural network specification are better than what we could obtain with a linear regression model, despite efforts to include in the regression model what we expected to be the most important nonlinearities and interaction terms.

The plot of actual and predicted values for the 3 node specification is presented in Figure 5. As seen here, the scatter is tighter than the regression result presented in Figure 4. Examination of the residuals verified that the observations with the largest errors were the same for both modeling approaches.

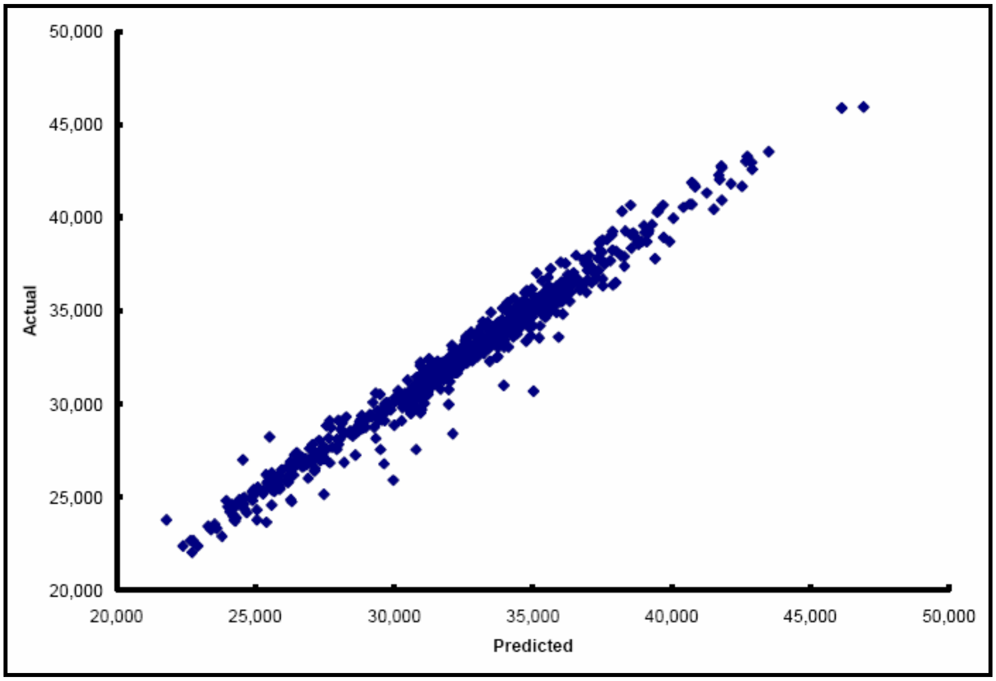

Figure 5: Actual Vs. Predicted Loads at 3 p.m. – Neural Network Model

In addition to the summary statistics presented above, the model residuals were examined for presence of autocorrelation. The Durbin-Watson statistic for the 3-node model was 1.64, indicating absence of first order autocorrelation.

Other Hours

The same method applied to other hours of the day provides a set of models that can be used to forecast next-day loads. Across hours, the in-sample and out-ofsample MAPE values range from a low of 0.98% to a high of 2.43%. To visualize the predictive power of this set of hourly models, actual and predicted values for the summer and winter peak weeks are shown in Figure 8 and Figure 9.

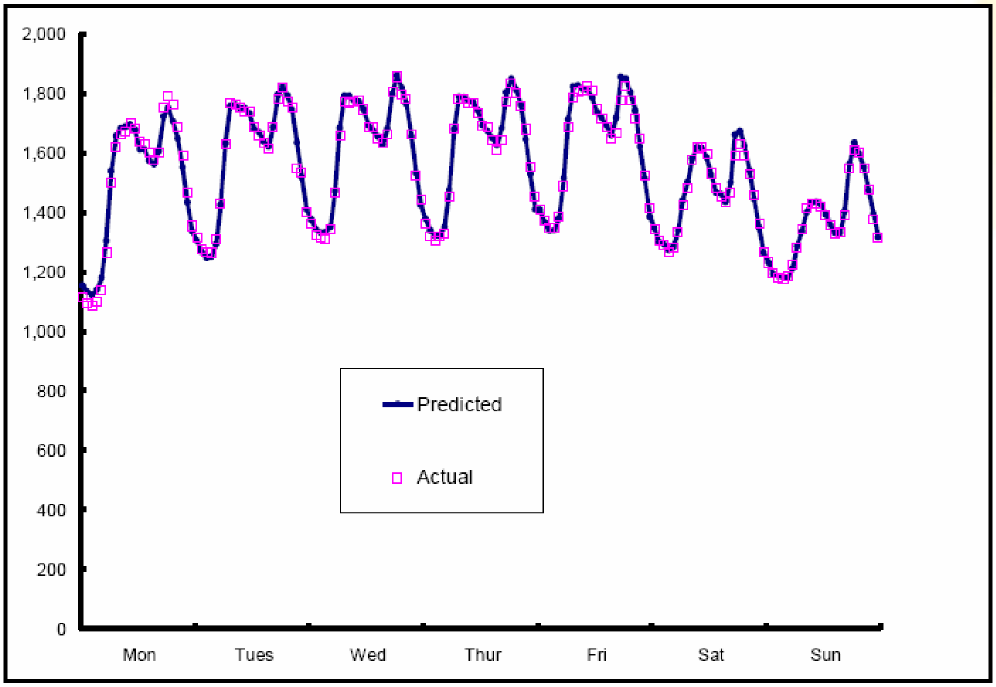

Figure 6: 1993 Summer Peak Week

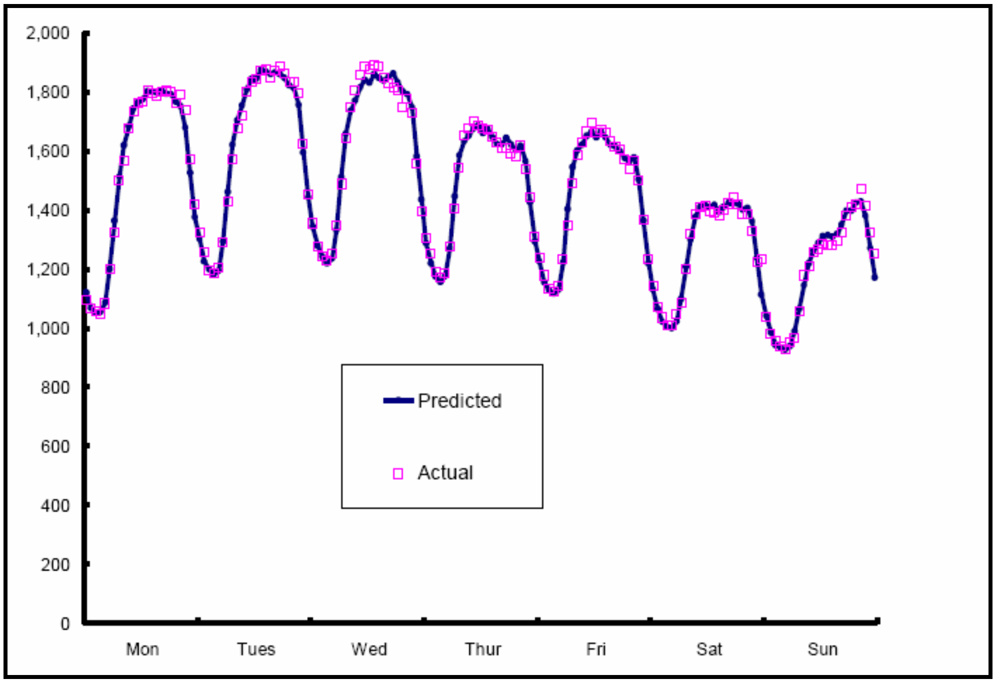

Figure 7: 1993 Winter Peak Week

5. Conclusion

Artificial neural networks provide a flexible nonlinear framework with many similarities to structural econometric models. These models are well suited to the forecasting task for data like the hourly load data analyzed here. However, there is a danger that the approach will be treated as a black-box, and something that is difficult to understand. As we have shown here, these models have much in common with standard econometric approaches, and can be discussed in terms that are familiar to individuals who are comfortable with structural econometric forecasting.

However, there is new ground here as well. The key difference is in the flexibility of these models. Estimation of parameters is a bigger task because the models are nonlinear and because the training exercise amounts to a search for a useful functional form. For a given number of nodes, training involves the search for a set of nonlinearities and interactions that provide the best model fit to the historical data. The objective function for error minimization is exceedingly complex, because there are a large number of solutions that are locally optimal and that fit the data well. As a result, it is necessary to search parameter space to find a good solution, one that works well both in sample and in reserved test periods. These solutions can also be evaluated and compared based on a variety of standard model statistics.

As long as the number of nodes is kept to a reasonable level, the result is a forecasting model that is powerful, robust, and sensible. This method is useful for forecasting and also for exploratory analysis that can be used to examine issues related to functional form. The results can be used directly, and they can also be used to strengthen econometric models through the identification of important nonlinearities and interactions.

References

1. Azoff, E. M. *"Neural Network Time Series Forecasting of Financial Markets,"* John Wiley & Sons, 1994.
2. Goffe, William L, Gary D. Ferrier, and John Rogers, "Global Optimization of Statistical Functions with Simulated Annealing." *Journal of Econometrics* 60 (1994), 65-99, North
3. Kuan, Chung-Ming and Halbert White, "Artificial Neural Networks, An Econometric Perspective." UCSD Discussion Paper, June 1992
4. Masters, Timothy (1995), Advanced Algorithms for Neural Networks, New York: *John Wiley & Sons*.
5. IMSL Stat/Library, Fortran Subroutines for Statistical Applications (1994). Boulder Colorado: *Visual Numerics, Inc.*
6. Khotanzad, Alireza, Rev-Chue Hwang and Dominic Maratukulam (1993). "Hourly Load Forecasting by Neural Networks." *Proceedings of the IEEE PES Winter Meeting*, February. Ohio.
7. Papalexopoulos, Alex D., Shangyou Hao and Tie-Mao Peng (1994). "An Implementation of Neural Network Based Load Forecasting Models for the EMS." *IEEE Transactions on Power Systems*, Vol. 9., No. 4 (November).
8. Rummelhart, D.E., B.E. Hinton, and R.J. Williams. "Learning Internal Representations by Error Propagation." In D.E. Rumelhart and J.L. McClelland (eds.) *Parallel Distributed Processing: Explorations in the Microstrucutres of Cognition*. MIT Press, Cambridge, Mass, 1986.
9. Robins., and S. Monro. "A Stochastic Approximation Method." *Annals of Mathematical Statistics 22* (1951), 400-407.
10. White, Halbert, "Neural-Network Learning and Statistics." *AI Expert*, December, 1989.

Author Biographies

J. Stuart McMenamin, Ph.D. Vice President at Itron, where he specializes in the fields of energy economics, statistical modeling, and software development. Over the last 20 years, he has managed numerous projects in the areas of system load forecasting, price forecasting, retail load forecasting, end-use modeling, regional modeling, load shape development, and utility data analysis. In addition to directing large analysis projects, Dr. McMenamin directs the development of Itron's forecasting software products. He has also directed the development of software packages for external clients, including the EPRI end-use models, regional economic forecasting packages, and home energy rating software. Dr. McMenamin received his B.A. in Mathematics and Economics from Occidental College and his Ph.D. in Economics from UCSD.

Frank A. Monforte, Ph.D. is Vice President of Forecasting at Itron, where he is a leading authority in the areas of short-term load forecasting, load profiling, retail scheduling, end-use forecasting, and statistical and mathematical modeling. Dr. Monforte directs the development, support, and implementation of Itron's forecasting and load profiling tools. In addition to his forecasting responsibilities, he is a nationally recognized authority in the area of industrial end-use analysis. Dr. Monforte has co-authored award-winning publications on a range of problems including the use of neural networks for short-term load forecasting, long-term end-use forecasting, and the use of nonlinear programming techniques for development of a least cost gas supply planning tool. Dr. Monforte received his B.A. in Economics from the University of California, Berkeley and his Ph.D. in Economics from the University of California, San Diego.

Itron Inc.

Itron is a leading technology provider and critical source of knowledge to the global energy and water industries. Nearly 3,000 utilities worldwide rely on Itron technology to deliver the knowledge they require to optimize the delivery and use of energy and water. Itron delivers value to its clients by providing industry-leading solutions for meter data collection, energy information management, demand response, load forecasting, analysis and consulting services, transmission and distribution system design and optimization, web-based workforce automation, C&I customer care, as well as enterprise and residential energy management.

To know more, start here: www.itron.com

Itron Inc.
Corporate Headquarters
2111 North Molter Road
Liberty Lake, Washington, 99019
U.S.A.
Phone: 1.800.635.5461
Fax: 1.509.891.3355

Itron Inc.
Energy Forecasting - West
11236 El Camino Real
San Diego, California, U.S.A. 92130-2660
Phone: 1.800.755.9585
Fax: 1.858.481.7550

Itron Inc.
Energy Forecasting - East
20 Park Plaza, Suite 910
Boston, Massachusetts 02116-4399
Phone: 1.617.423.7660
Fax: 1.617.423.7664