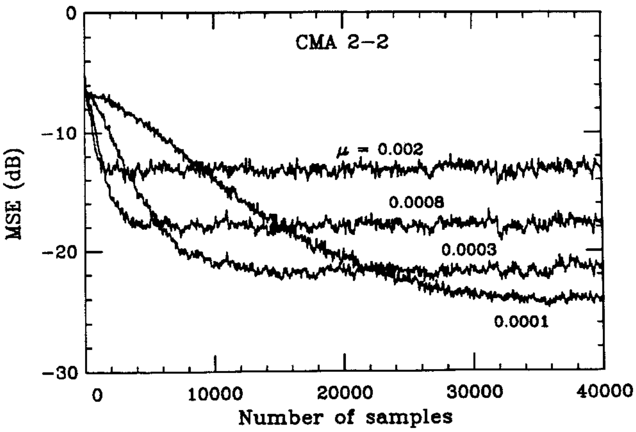
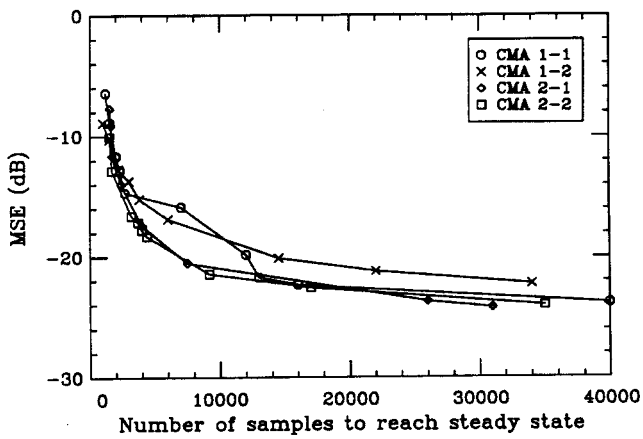


A



B

Figure 5. *A*, CMA learning curves. *B*, CMA performance curves. (From Shynk, J. J., Gooch, R. P., Giridhar, K., and Chan, C. K., 1991, A comparative performance study of several blind equalization algorithms, in *Proceedings of the SPIE Conference on Adaptive Signal Processing*. © 1991, SPIE. Reprinted with permission.)

variations of the LMS algorithm, including those that have less complexity or improved convergence properties. For example, CMA is a blind stochastic-gradient algorithm that can be used instead of the LMS algorithm when an explicit training sequence is not available. The recursive-least-squares (RLS) algorithm is an adaptive algorithm based on the method of least squares that offers faster convergence rates (compared with the LMS algorithm), but at the expense of an increased computational complexity (Haykin, 2002).

The adaptive filter configuration described in this article is the basic component of a multilayer perceptron. These additional layers provide greater nonlinear modeling capabilities, which is usually necessary for complex applications such as speech and image processing. Stochastic-gradient algorithms are typically used to adjust the weights of a multilayer perceptron. They are similar to the adaptive algorithms described in this article, but they have an additional degree of complexity owing to the cascade of layers. One such algorithm, known as the backpropagation algorithm (Rumelhart and McClelland, 1986), has been successfully applied to a number of signal processing problems (Widrow and Lehr, 1990).

Road Map: Applications
Background: Perceptrons, Adalines, and Backpropagation
Related Reading: Forecasting; Kalman Filtering: Neural Implications; Recurrent Networks: Learning Algorithms

References

Giannakis, G. B., Ed., 1999, Highlights of signal processing for communications, *IEEE Signal Proc. Mag.*, 16:14–50. ◆

Godard, D. N., 1980, Self-recovering equalization and carrier tracking in two-dimensional data communication systems, *IEEE Trans. Commun.*, COM-28:1867–1875.

Gooch, R. P., and Lundell, J. D., 1986, The CM array: An adaptive beamformer for constant modulus signals, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, New York: IEEE, pp. 2523–2526.

Haykin, S., 1999, *Neural Networks: A Comprehensive Foundation*, Upper Saddle River, NJ: Prentice-Hall. ◆

Haykin, S., 2002, *Adaptive Filter Theory*, 4th ed., Upper Saddle River, NJ: Prentice-Hall. ◆

Lippmann, R. P., 1987, An introduction to computing with neural nets, *IEEE ASSP Mag.*, 4:4–22. ◆

Ljung, L., and Söderström, T., 1983, *Theory and Practice of Recursive Identification*, Cambridge, MA: MIT Press.

Proakis, J. G., 2001, *Digital Communications*, 4th ed., New York: McGraw-Hill. ◆

Qureshi, S. U. H., 1985, Adaptive equalization, *Proc. IEEE*, 73:1349–1387. ◆

Rumelhart, D. E., and McClelland, J. L., Eds., 1986, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Cambridge, MA: MIT Press. ◆

Shynk, J. J., and Chan, C. K., 1993, Performance surfaces of the constant modulus algorithm based on a conditional Gaussian model, *IEEE Trans. Signal Proc.*, 41:1965–1969.

Shynk, J. J., Gooch, R. P., Giridhar, K., and Chan, C. K., 1991, A comparative performance study of several blind equalization algorithms, in *Proceedings of the SPIE Conference on Adaptive Signal Processing*, Bellingham, WA: SPIE, pp. 1565:102–117. ◆

Treichler, J. R., and Agee, B. G., 1983, A new approach to multipath correction of constant modulus signals, *IEEE Trans. Acoust. Speech Signal Proc.*, ASSP-31:459–472.

Widrow, B., and Lehr, M. A., 1990, 30 years of adaptive neural networks: Perceptron, madaline, and backpropagation, *Proc. IEEE*, 78:1415–1441. ◆

Widrow, B., and Stearns, S. D., 1985, *Adaptive Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall. ◆

Forecasting

Lyle H. Ungar

Introduction

Forecasting the future values of sequences of observations is, in many ways, ideally suited for neural networks. Large amounts of data may be available, and the underlying relationships are often nonlinear and unknown. Neural nets, mostly of the standard backpropagation type (see Backpropagation: General Principles), have been used with great success in many forecasting applications, including forecasting electricity load, freeway traffic volume, solar cycles, milk yields, tourism demand, grain drying times, ambient air quality, exchange rates, inflation, unemployment, disease epidemics, fish stock levels, sea surface temperatures, sales volumes, flood occurrence in Moravia, and rainfall in Bangladesh. However, in not all of such cases do neural networks outperform conventional ARMA models. This article looks at the use of neural nets for forecasting, with particular attention to understanding when they perform better or worse than other technologies.

The success of neural networks in forecasting depends significantly on the characteristics of the process being forecast. One may want to predict minute-by-minute progress of a chemical reaction, hour-by-hour power usage (load) for an electric power utility, daily weather, monthly prices of products and inventory levels, and quarterly or yearly sales and profits. These problems differ in the quantity and type of information available for forecasting, and hence call for different forecasting techniques. One also needs to choose an appropriate network architecture.

Forecasting problems can be characterized on a number of dimensions: (1) Is a single series of measurements used, as is often done in conventional forecasting, or are multiple related measurements available? (2) Are the data seasonal or not? Monthly or quarterly data such as sales volume or energy use often show strong seasonal variation, while annual data or data measured each second or minute do not. (3) The number of observations and (4) the degree of randomness (signal/noise ratio) of the process also strongly limit the complexity of the model that can be fit. If data are only available annually for the past 10 or 20 years, and if no measurement is available for most of the disturbances, one should not expect to be able to fit a complex model such as a neural network. This is unfortunately the case for many forecasting problems such as those represented in the Makerdakis collection (described below). (5) Finally, for some forecasting problems, one only requires prediction a single time step in the future, while for others, multiple time step forecasts are required. This has implications for the method used to train the neural network.

Before looking at neural networks, we will briefly review conventional forecasting methods. Forecasting has mostly been done using one of two different classes of methods, depending on whether the data are seasonal or not. For monthly data, such as sales or unemployment levels, the seasonal variation is often removed by dividing the series by an index representing the historical seasonal variation. For example, dividing the unemployment rate for each month (perhaps averaged over several years) by the average annual unemployment rate gives an index that indicates monthly variations. This index will have an average value of one. Dividing the actual unemployment rate in a given month by the index for that month gives the seasonally adjusted unemployment rate, which shows overall trends after typical monthly variations are accounted for. A linear or exponential regression (i.e., fitting the data as a linear or exponential function of time), or some form of smoothing such as a moving average, can then be used to make predictions of the deseasonalized unemployment. Actual levels are then forecast by multiplying these base predictions by the index for the month being forecast (Makridakis, Wheelwright, and McGee, 1983).

In contrast, for many complex processes such as chemical plant production, robots, or stock prices, the best prediction of the near future is obtained by using an appropriately weighted combination of recent measurements of the variable being predicted and other correlated variables. The most widely used approach is the Auto-Regressive Moving Average (ARMA) model. For example, to predict the value of a variable y (such as a temperature or a pressure or a stock price) at time $t + 1$ using past values of y and of a second variable z, one would use a linear regression to fit a model of the form

$$y_{t+1} = c_0 + c_1 y_t + c_2 y_{t-1} + c_3 y_{t-2} + \cdots + c_n z_t + c_{n+1} z_{t+1} + \cdots \quad (1)$$

Note that ARMA models differ from the linear regression models mentioned above in that they are functions of previous variables rather than of time.

Neural networks can be used to learn a nonlinear generalization of ARMA models of the form

$$y_{t+1} = f(y_t, y_{t-1}, y_{t-2}, \ldots, z_t, z_{t+1}, \ldots) \quad (2)$$

When the process is nonlinear and sufficient data are available, the neural networks will provide a more accurate model than the linear ARMA model. See Box and Jenkins (1970) for extensive descriptions of conventional ARMA models and the Box-Jenkins modeling approach, which involves picking a model of the form of Equation 1 with some subset of the coefficients set to zero. Later in this article we summarize the results of a number of studies that compare ARMA and neural network models.

Two other modeling methods are also often used by engineers, Kalman filtering and Wiener-Voltera series. Kalman filters (see KALMAN FILTERING: NEURAL IMPLICATIONS) assume a known model structure in which the parameters and their covariance, which is modeled explicitly, may be changing over time. Kalman filters are good for modeling relatively simple but noisy processes, but, unlike neural networks, they do not form nonparametric models that can accurately forecast the behavior of nonlinear systems. Wiener-Voltera series are polynomial expansions fitted to past data. As such, they, like neural nets, can approximate arbitrary functions. However, for models with multiple inputs they require more data than neural networks to obtain an equal level of accuracy.

Using Neural Nets for Forecasting

Neural networks are most often used to fit ARMA-style models of raw time series data from one or more measurements, but they can also be used as a piece of larger forecasting systems, such as in combination with deseasonalizing (i.e., forecasting a time series from which the seasonal component has been removed, as described above). Even for the simpler ARMA-style models, attention to the method is required if one is making forecasts multiple time steps in the future rather than a single time step.

Direct Versus Recurrent Prediction

A simple form of multistep forecasting is direct prediction (Figure 1*A*), in which a network takes past values as inputs and has separate outputs for predictions one, two, and more time steps in the future. Alternatively, one can train a network to predict one time step in the future and then use the network recursively to make multistep predictions (Figure 1*B*). Such networks are sometimes called *externally recurrent networks*, in contrast to networks that have internal memory. Direct forecasting networks are easier to build than externally recurrent nets because they do not require unfolding in time (described below), but the predictions are generally less accurate, since they have more parameters that must be fit from the same limited data.

The obvious way to train a network such as is used in Figure 1*B* is to minimize the error on the one-time-step predictions. Unfortunately, this does not give optimal networks for multistep predictions. To better understand this somewhat confusing point, consider the case of a simple linear ARMA model:

$$y_{t+1} = c_0 + c_1 y_t + c_2 y_{t-1} \quad (3)$$

A two-step-ahead prediction would then take the form

$$y_{t+2} = c_0 + c_1(c_0 + c_1 y_t + c_2 y_{t-1}) + c_2 y_t \quad (4)$$

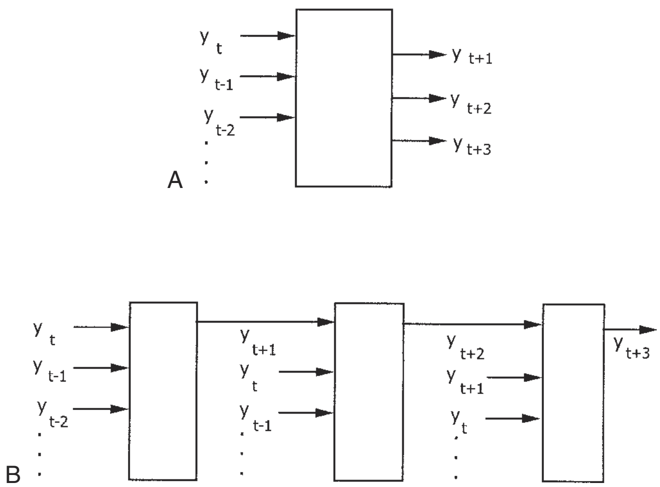


Figure 1. *A*, Direct prediction using a neural network. *B*, Recurrent one-step-ahead prediction using a neural network repeatedly.

Selecting coefficients c_0, c_1, and c_2 to minimize the prediction error for the one-step-ahead error yields a different equation than selecting the same coefficients to minimize the error in the two-step prediction. (Note that the former is a linear regression problem, whereas the latter requires nonlinear regression because the coefficients multiply each other.) More accurate long-range predictions are obtained by training to minimize the multistep prediction error. The solution using backpropagation uses the same unfolding in time or other solution methods as for internally recurrent networks (see RECURRENT NETWORKS: LEARNING ALGORITHMS). This and related issues are covered in detail in books on conventional system identification methods (e.g., Ljung and Torsten, 1983). Much good work has been done using recurrent nets to model time series (e.g., Mozer, 1994).

Combining Neural Networks with Other Methods

There are a number of ways in which neural networks can be combined with data preprocessing techniques, first principles (mechanistic) with partial models of the process being forecast, and with other forecasting techniques. Most commonly, if there is a strong seasonal component to the data, the data may be deseasonalized and the neural net used to forecast the basic trend. It may appear pointless to use a seasonal index when it is well known that neural networks can approximate arbitrary functions, which should include any seasonal variation. Experience indicates that if sufficient data are available, this is true, but that for shorter time series, deseasonalizing gives more accurate forecasts.

Similarly, when modeling complex physical systems, much better forecasts can be obtained with much less data when prior knowledge (e.g., in the form of mass, energy, or kinematic constraints on the variables, or in terms of monotonic relations between measured and forecast variables) is built into the network (Psichogios and Ungar, 1992). In a typical example, the equations governing a fermentation reactor are known except for the growth kinetics of the cells (e.g., yeast) in the reactor. If a neural network is used just to approximate the growth kinetics rather than to model the whole system, models are learned that are more accurate and that extrapolate better to operating regimens where no data are available. Such hybrid or "gray box" methods are popular in science and engineering.

Neural networks can also be used in conjunction with conventional forecasting methods. For example, one can often produce more accurate forecasts by providing several conventional forecasts as input to the neural network. In this case, the network serves partly as a combining method in which the network produces a weighted average of the different forecasts (Foster, Collopy, and Ungar, 1992). Such combining of forecasts is widely practiced in the forecasting community, mostly with relatively arbitrary combining weights.

Assessing Neural Nets for Forecasting

There are several difficulties in assessing forecasting methods. The most serious is that the results of a single forecast tell little about whether the method will be superior for other forecasts. In testing any method, it is important to have a large set of representative time series on which the methods will be tested. An example of such a collection of time series that has been widely used to compare forecasting methods is the Makridakis competition, or M-competition, model (Makridakis et al., 1982). This competition included 1,001 series and evaluated 24 forecasting methods. The series were taken from a variety of organizations in a number of countries and included macroeconomic, microeconomic, industrial, and demographic data such as production levels, net sales, unemployment, spending, GNP, vital statistics, and infectious disease incidence. The series included yearly, quarterly, and monthly series, but no series arising from securities or commodities trading. These time series all involve only a single variable and do not provide correlated variables, which might enhance the predictions.

One must also decide which error criteria to use. The most obvious criterion, and the one that is optimized by standard neural networks, is minimization of the mean squared prediction error. This criterion has the property that a small number of unusual series may have a large effect on the error. In looking at combined errors for different time series, one must, of course, also normalize for the different magnitudes of the series. Thus, forecasters often measure performance by using measures that are more robust to outliers or atypical time series.

Three error measures that have proved particularly robust are the percentage of time a method had a lower absolute error than the "no-change" forecast (or "percent better"), the relative absolute error (or RAE), and the median absolute percent error (or mdAPE). The RAE is calculated as the geometric mean across all series i of

$$\mathrm{RAE}_i = \frac{\sum_{t=1}^{T} |\tilde{x}(t) - x(t)|_i}{\sum_{t=1}^{T} |x(0) - (t)|_i} \tag{5}$$

where $\tilde{x}(t)$ is the forecast and $x(t)$ represents the true value of the series at time t. The RAE represents a comparison over the forecast horizon T for series i of the absolute error of the forecast method, compared to the no-change or random walk forecast. One then calculates a geometric mean over all the series:

$$\mathrm{RAE} = \left[\prod_{i=1}^{n} \mathrm{RAE}_i\right]^{1/n} \tag{6}$$

The median average percent error is defined as the median across all series i of

$$\mathrm{APE}_i = \frac{1}{T}\sum_{t=1}^{T} 100\,\frac{|\tilde{x}(t) - x(t)|_i}{|x(t)|_i} \tag{7}$$

Good forecast performance is reflected in higher "percent betters" and lower RAEs and mdAPEs.

In assessing neural networks for forecasting, one must compare the accuracy of the neural networks with that of other statistical tools such as exponential smoothing (for a single time series) or linear ARMA models (for several correlated time series). Surprisingly, many studies fail to compare neural network forecasts with well-made conventional forecasts.

Table 1 lists some applications in which neural networks have been used for forecasting. Almost all of the studies used standard backpropagation networks with less than a dozen inputs and less than a dozen hidden nodes, with the exact architecture being selected by trial and error. Also, most of the studies used data from a single source, and most of the authors evaluated their results on the basis of the mean squared error on out-of-sample forecasts (i.e., error when forecasting data other than that used for building the model). Table 1 does not include any studies using chaotic time series such as from the Mackey-Glass equation, which give little insight into neural network forecasts of realistic data. See Vemuri and Rogers (1994) for a good collection of reprints of a wide variety for neural network forecasting studies, including all studies cited in Table 1 that are not listed in the references. There is also an extensive literature on neural network forecasting for process control (see PROCESS CONTROL in the First Edition). Process control and robotics applications have seen some of the most successful use of neural networks for forecasting, as the processes involved are often sufficiently multivariable and nonlinear to warrant the use of neural networks but sufficiently well characterized and free of noise to allow accurate models to be built.

Dangers in Using Forecasts

Forecasts rely on a number of assumptions. They assume that the system that is modeled remains constant, i.e., that the model that held when the model was built still applies when the forecast is made. If the system structure is evolving over time, techniques from adaptive control may be more appropriate. It is also implicitly assumed when forecasting using neural networks with multiple inputs that the covariance structure of the inputs will remain constant. This presents a major difficulty when modeling systems that have feedback in them, if the feedback structure is variable. For example, consider a house controlled by a thermostat. One will typically find that the heater will be on more often when the house is cold (this is, after all, what the heating system is designed to do). Forecasts of future house temperature can be accurately made using historical temperature measurements. If, however, these forecasts are used as part of the control scheme (the thermostat), then instability often results, since the forecasts fail to account for the new thermostat behavior. Similar situations often occur in economics and marketing, where forecasts can result in new laws being passed or in new prices being charged (and resulting actions by competitors), thus invalidating the original forecast. Unfortunately, there is generally little that one can do other than monitoring forecasts and distrusting them or collecting more data, if the process being forecast changes. (This is true in linear regression as well, where it is impossible to tell which of two highly correlated inputs is responsible for changes in an output, but at least one can easily detect the problem in linear problems by examining the uncertainty on the regression coefficients, whereas it is usually concealed in neural nets.)

Table 1. Forecasting Using Neural Nets: Sample Results

Application	Authors	Results	Compared with
Car sales, airline passengers	Tang et al.	NNet better for longer-term forecast; Box-Jenkins better for shorter	Box-Jenkins
Currency exchange rates	Weigend et al.	NNets better	Random guessing
Electric load forecasting	Park et al.	NNet better	Currently used technology (unclear what)
Electrochemical reaction	Hudson et al.	Prediction looks good	—
Flour prices	Chakraborty et al.	NNets better than ARMA	ARMA
Polypropylene sales	Chitra	NNets slightly better than ARMA	ARMA
Stock prices	White	NNets provide no benefit	Random walk
Widely varied (Makridakis collection)	Foster et al.	NNets better on quarterly data, worse on annual data	Many exponential smoothing and deseasonalizing methods

Discussion

Neural networks have many demonstrated successes as forecasting tools and a smaller number of documented failures. All the usual warnings about model building apply. In particular, to build a good model, one needs good data. When the data are noisy and occur in short series, neural networks often fail to do better than simple forecasting techniques. For example, the 181 yearly series of the M-competition, which have a mean length of 19 data points on which to base a prediction, do not provide a good basis for complex nonlinear models. Neural networks generally give significant improvements over conventional forecasting methods when applied to monthly data in the M-competition set but not when applied to yearly data (Hill, O'Connor, and Remus, 1996). This is probably due to the high ratio of noise to data in the yearly data.

It may also be the case that the data are truly random or that the key independent variables are not being measured. Research suggests that this is true of the stock market (White, 1988). If this is true, then neural networks will not produce useful market forecasts, although they may help sell forecasting products. Several fund managers claim that they are getting superior predictions using neural networks, but for obvious competitive reasons, they generally do not provide enough information to test the claims. Moody (1998) provides a good discussion of the issues in forecasting the economy.

Neural networks have proved successful in a number of applications such as forecasting prices (Chakraborty et al., 1992), product demand (Chitra, 1993), electric utility loads (Yu, Moghaddamjo, and Chen, 1992), and inventory levels (see Table 1). Such problems are characterized by ample measurements with a relatively high signal-to-noise ratio. In most cases, substantially better performance is obtained by using several related inputs to the network. For example, in forecasting wheat prices in three cities, superior performance was found by using recent wheat prices and measures of the local earning power. Similarly, in forecasting demand for polypropylene production, several macroeconomic variables were fed into the network. On longer, more deterministic time series, such as measuring the progress of a chemical reaction, neural networks have been shown to be a relatively accurate means of forecasting even chaotic series (Hudson et al., 1990; Lapedes and Farber in Vermuri and Rogers, 1994).

All of the applications cited above use standard backpropagation networks, occasionally with some degree of structure built into the network. For example, in the currency exchanges, excess weights were eliminated, while for forecasting wheat prices, past values of prices in three different cities were used to predict the logarithm of flour prices. All have demonstrated better performance than conventional forecasting methods, except when only short time series were available (10 to 30 data points) or when it was unclear if there was an underlying model other than a biased random walk (e.g., stock prices). However, the gains in accuracy over conventional forecasting methods are often relatively small, and overfitting is a common problem. Many companies are now using neural networks for problems such as demand forecasting. When sufficient data are available and care is taken to avoid overfitting, neural networks work well.

Road Map: Applications
Related Reading: Kalman Filtering: Neural Implications; Recurrent Networks: Learning Algorithms

References

Box, G., and Jenkins, G., 1970, *Time Series Analysis: Forecasting and Control*, San Francisco: Holden-Day. ◆

Chakraborty, K., Mehrotra, K., Mohan, C. K., and Ranka, S., 1992, Forecasting the behavior of multivariate time series using neural networks, *Neural Netw.*, 5:961–970. ◆

Chitra, S. P., 1993, Use neural networks for problem solving, *Chem. Eng. Prog.*, April, pp. 44–52. ◆

Foster, B., Collopy, F., and Ungar, L. H., 1992, Neural network forecasting of short noisy time series, *Comput. Chem. Eng.*, 16:293–298. ◆

Hill, T., O'Connor, M., and Remus, W., 1996, Neural network models for time series forecasts, *Manage. Sci.*, 42:1082–1092.

Hudson, J. L., et al., 1990, Nonlinear signal processing and system identification: Applications to time series from electrochemical reactions, *Chem. Eng. Sci.*, 45:2075–2981.

Ljung, L., and Torsten, S., 1983, *Theory and Practice of Recursive Identification*, Cambridge, MA: MIT Press.

Makridakis, S., et al., 1982, The accuracy of extrapolation (time series) methods: Results of a forecasting competition, *J. Forecast.*, 1:111–153.

Makridakis, S., Wheelwright, S., and McGee, V., 1983, *Forecasting: Methods and Applications*, New York: Wiley. ◆

Moody, J., 1998, Forecasting the economy with neural nets: A survey of challenges and solutions, *Lecture Notes Comput. Sci.*, 1524:347–371.

Mozer, M. C., 1994, Neural net architectures for temporal sequence processing, in *Time Series Prediction* (A. S. Weigend and N. A. Gershenfeld, Eds.), Menlo Park, CA: Addison-Wesley, pp. 243–264.

Psichogios, D. C., and Ungar, L. H., 1992, A hybrid neural network: First principles approach to process modeling, *Am. Inst. Chem. Eng. J.*, 38:1499–1512.

Vemuri, V. R., and Rogers, R. D., 1994, *Artificial Neural Networks: Forecasting Time Series*, Los Alamitos, CA: IEEE Computer Society Press. ◆

White, H., 1988, *Economic Prediction Using Neural Networks: The Case of IBM Daily Stock Returns*, in *Proceedings of the IEEE International Conference on Neural Networks*, San Diego, p. II-451.

Yu, D. C., Moghaddamjo, A. R., and Chen, S.-T., 1992, Weather sensitive short-term load forecasting using a nonfully connected artificial neural network., *IEEE Trans. Power Syst.*, 7:1098–1105.

Gabor Wavelets and Statistical Pattern Recognition

John Daugman

Introduction

Starting around 1960, for about three decades investigation into the functioning of the mammalian primary visual cortex was dominated by recordings from single neurons. Using relatively simple stimuli such as oriented bars of light (e.g., Hubel and Wiesel, 1962, 1974), the apparent coding dimensions underlying spatial vision were mapped out by measuring tuning curves of individual neural responses as functions of stimulus parameters. Although methods later moved on, with innovations such as population recordings, noninvasive imaging with photovoltaic dyes, and novel anatomical techniques, the single-unit recording paradigm left a rich legacy of data that lent itself to modeling in engineering terms such as filtering, feature extraction, transform coding, and dimensionality reduction.

In this framework, the key functional concept is that of a neuron's *receptive field*, which specifies that region of two-dimensional (2D) visual space in which image events or structure can influence the neuron's activity. More exactly, the neuron's *receptive field profile* indicates the relative degree to which the cell is excited or inhibited by the distribution of light as a function of its spatial position within the receptive field. Through careful measurements with precisely defined stimuli, the receptive field profile of a *linear* neuron (one obeying proportionality and superposition in its responses to stimuli) reveals how it will respond to *any* pattern and allows the neuron to be analyzed in signal processing terms as a filter. The powerful mathematical tools of linear systems analysis (including Fourier analysis) are the basis of such extrapolations, subject always to the assumption of linearity. More recent findings of adaptive, nonlinear, remote interactions between visual neurons "beyond the classical receptive field" undermine the linear filter perspective and may even call into question the whole notion that a neuron has a stable receptive field profile. Nevertheless, impressive practical results have been achieved in engineering applications of one such model inspired by the classical receptive field data. This article reviews the model that has come to dominate the classical description of cortical simple cells and their inputs to complex cells, and it reviews some successful applications of that scheme within computer vision and statistical pattern recognition.

Receptive Fields and 2D Gabor Wavelets

Typical two-dimensional receptive field profiles of simple cells in the feline visual cortex (Jones and Palmer, 1987) are shown in the top row of Figure 1. There are arguably five major degrees of freedom (i.e., independent forms of variation) spanned by the spatial receptive field structure of such neural populations. These can be regarded as defining the dimensions of the spatial visual code at this cortical level. The first two degrees of freedom are the *location* of a neuron's receptive field, defined by retinotopic coordinates (x, y). The third is the *size* of its receptive field (which can be described using a single scalar diameter, provided we view variation in the field width/length aspect ratio as a secondary population structure). The fourth is the *orientation* of the boundaries separating excitatory and inhibitory regions, as seen in Figures 1 and 3, normally also corresponding to the direction of receptive field elongation. The fifth is the *symmetry*, which may be even or odd, or some linear combination of these two canonical states. (Any function can be decomposed into the sum of an even function plus an odd function, and their relative amplitudes define a continuum that allows this fifth dimension to be regarded as *phase*.)

These degrees of freedom in the spatial visual code also correspond to certain dimensions of the "cortical architecture" (rules of topographic and modular organization), although such structure is