Spectral Pattern Recognition Using Self-Organizing MAPS

Barry K. Lavine,* Charles E. Davidson, and David J. Westover

Department of Chemistry, Clarkson University, Potsdam, New York 13699-5810

Received December 5, 2003

A Kohonen neural network is an iterative technique used to map multivariate data. The network is able to learn and display the topology of the data. Self-organizing maps have advantages as well as drawbacks when compared to principal component plots. One advantage is that data preprocessing is usually minimal. Another is that an outlier will only affect one map unit and its neighborhood. However, outliers can have a drastic and disproportionate effect on principal component plots. Removing them does not always solve the problem for as soon as the worst outliers are deleted, other data points may appear in this role. The advantage of using self-organizing maps for spectral pattern recognition is demonstrated by way of two studies recently completed in our laboratory. In the first study, Raman spectroscopy and self-organizing maps were used to differentiate six common household plastics by type for recycling purposes**.** The second study involves the development of a potential method to differentiate acceptable lots from unacceptable lots of avicel using diffuse reflectance near-infrared spectroscopy and self-organizing maps.

INTRODUCTION

Scientists often use graphical methods to characterize their data. If there are only two or three measurements per sample, the data can be plotted directly. By examining the plot, a scientist can search for similarities and dissimilarities among data points and find natural clusters in the data. When each sample is represented by n measurements ($n > 3$), then a two- or three-dimensional representation of the measurement space is needed to visualize the relative position of the data points in n-space. This representation should accurately reflect relationships that exist between sample points in the original n-space. One approach that is used by scientists for representing data in an n-dimensional measure space is a mapping and display technique called principal component analysis.[1,2]

Principal component analysis is a method for transforming the original measurement variables into new, uncorrelated variables called principal components. Each principal component is a linear combination of the original measurement variables. Using this procedure, a set of orthogonal axes that represent the direction of greatest variance in the data is found. If the original measurement variables are correlated, only two or three principal components are often necessary to explain the information present in the data. Principal component analysis has been applied to spectral data for dimensionality reduction, to identify outliers, display structure, and classify samples.[3−10]

Another approach for representing data points in an n-dimensional measure space involves using an iterative technique known as the Kohonen neural network[11,12] or self-organizing map (SOM). A Kohonen neural network maps multivariate data onto a layer of neurons arranged in a two-dimensional grid. Each neuron in the grid has a weight associated with it, which is a vector of the same dimension as the pattern vectors comprising the data set. A pattern vector is passed to each neuron in the network, and the neuron whose weight vector is the most similar to the pattern vector is declared the "winner". For the winning neuron, the weight vector is adjusted to more closely resemble the pattern vector. Neurons that surround the winning neuron are adjusted as well but to a lesser degree. This process, when completed, causes similar data vectors to respond to neurons that are near each other enabling the neural network to learn and display the topology of the data.

The procedure for implementing the Kohonen neural network is as follows. First, the network is initialized. The components of each weight vector are assigned random numbers. Training is performed by presenting the data one pattern at a time to the network. Because competitive learning is used, the pattern vector or sample is assigned to the neuron whose weight vector is closest, which is why the Euclidean distance is computed between the pattern vector and each weight vector in the network. The nearest neuron is declared the winner, and the weight vector of the winning or central neuron and its neighbors are adjusted to more closely resemble the sample. Equation 1 gives the update rule

$$w_i(t+1) = w_i(t) + \eta(t) * \alpha(d_{ic})(x_i - w_{i,old}) \qquad (1)$$

where $w_i(t+1)$ is the ith weight vector for the next iteration, $w_i(t)$ is the ith weight vector for the current iteration, $\eta(t)$ is the learning rate function, $\alpha(d_{ic})$ is the neighborhood function, and x_i is the sample vector currently passed to the network. During a single epoch or iteration, the comparison of all sample data vectors to all weight vectors in the network and the modification of those weights will occur.

The learning rate is chosen by the user as a positive real number less than 1. Its value usually decreases during training. The neighborhood function determines the magnitude of the weight adjustment. This is based on d_{ic}, the link

* Corresponding author phone: (315)268-2394; fax: (315)268-6670; e-mail: bklab@clarkson.edu.

10.1021/ci030039y CCC: $27.50
Published on Web 04/23/2004

distance in the grid between the winning neuron and the neuron currently being adjusted. The magnitude of this adjustment is inversely proportional to the distance between the neuron undergoing adjustment and the winning neuron. The decrease of the neighborhood can be scaled to be linear with time, thereby reducing the number of neurons around the winner being adjusted during each epoch.

In the initial iterations, when the learning rate is high and the neighborhood is wide in scope, the neurons order themselves globally in relation to each other in the data. As the learning rate decreases and the size of the neighborhood diminishes, the passing of each sample to the network results in a relatively small change of only a few neurons. Over time, the neurons converge to an approximation of the probability distribution of the data. A large number of iterations are often required if the weight vectors are to converge toward a good approximation of the more numerous sample vectors.

Because similar data vectors will excite neurons that are near each other, the Kohonen neural network is able to develop a map of the data that preserves the topology of the original measurement space. Visual inspection of the map allows the user to identify outliers and recognize areas where groups of similar samples have clustered. An advantage of SOMs is that outliers affect only one map unit and its neighborhood, with the rest of the display available for investigation of the remaining data. Since the neurons are more likely to characterize areas of largest density, a single outlier will usually have little effect on the ending weights. By comparison, outliers often have a disproportionate effect on principal component plots because of the least squares property of principal components. A sample distant from the center of the data can pull the principal components toward it and away from the directions of "true" maximum variance, causing the remaining data to be compressed into a very small region of the map. Removing the outlier does not always solve the problem for as soon as the worst outliers are deleted other data points may appear in this role.

To better understand the advantages of SOMs for spectral pattern recognition, two studies were recently undertaken in our laboratory to evaluate the efficacy and efficiency of this mapping and display technique. The first study involved the use of Raman spectroscopy and self-organizing maps to differentiate six common household plastics by type for recycling purposes. In the second study, diffuse reflectance near-infrared spectroscopy and self-organizing maps were used to develop a potential method to differentiate acceptable from unacceptable lots of PH102 avicel. These two studies highlight the advantages and limitations of self-organizing maps and principal component analysis for spectral pattern recognition.

SOFTWARE IMPLEMENTATION

Somproj. A software system called PCKaNN,[13,14] which was developed in MATAB, was used to generate the principal component plots and SOMs used in the two studies. MATLAB was an obvious language choice for the development of PCKaNN. A powerful, functional, and easy to use language, MATLAB can accomplish tasks comparable to any lower level language, but in fewer lines of code. As a result, it is easier to translate ideas into working programs.

The module somproj, which has options for the control parameters of the SOM, was used in these two studies. The control parameters include the number of epochs (iterations), grid topology and size, the neighborhood function, the neighborhood adjustment factor, the learning rate function, and the learning rate. Figure 1 gives the on-screen layout of somproj. The SOM is a grid of neurons each assigned a class label based on the single closest sample to each neuron. The color of each neuron is determined by all samples responding to a neuron. If half the samples are from class 1 (red) and half are from class 2 (green) for a neuron, the corresponding neuron will have a color of yellow. Clicking on a neuron displays the neuron number, the samples that excite the neuron, and the distance of each sample from the neuron.

Initialization. The weights were initialized using a normal distribution with a mean of zero and a standard deviation of 0.1. This initialization scheme would allow us to work with mean centered and autoscaled data. Because all of the inputs to the Kohonen neural network are positive real values in the two studies shown, one might expect that initializing the weights using a uniform distribution from 0.0 to 0.1 would be the preferred scheme. However, changing the initialization scheme to uniformly distributed weights would not have significant impact on the results of these two studies and was therefore not pursued.

Since the goal is often to compare different feature subsets, it is important that comparisons be made with the same starting weights. Otherwise, even the same set of network parameters will give a different mapping of the data. Therefore, somproj initializes the weights when it starts and uses that same set of initial weights for all subsequent runs during a session.

Mapping Error. While the literature recommends training a neural network for thousands of epochs, our experience has shown that far fewer epochs are required when analyzing spectral or chromatographic data for clustering. For these types of problems, there is no guarantee that more epochs will produce a better mapping. A run with a large number of epochs can be just as susceptible to twists, kinks, and other distortions in the grid of neurons. Regardless of network parameters, it is important to be able to determine if these distortions exist. Within the Project software, there are three ways this can be done. The first method is to call the function sompco, which combines the data and weight matrices, finds the Euclidean distance matrix, and then calculates the principal coordinates which correspond to principal components. The display shows the location of each neuron in the principal component space of the data. While this is a good method for determining the local arrangement of the neurons, it is difficult to ascertain the global topology of the net.

The second method is to view the development of the weights during a run of the neural network. This method produces a principal component plot of the weights, showing the grid connection between neurons. The frequency with which this plot is updated is a user controlled variable. Depending on the value, the network can be displayed every epoch or simply viewed after training is complete. The third method involves superimposing the data on the principal component plot of the weights. Figure 2 shows several of these principal component plots after training for a simulated data set, which was generated from two multivariate normal

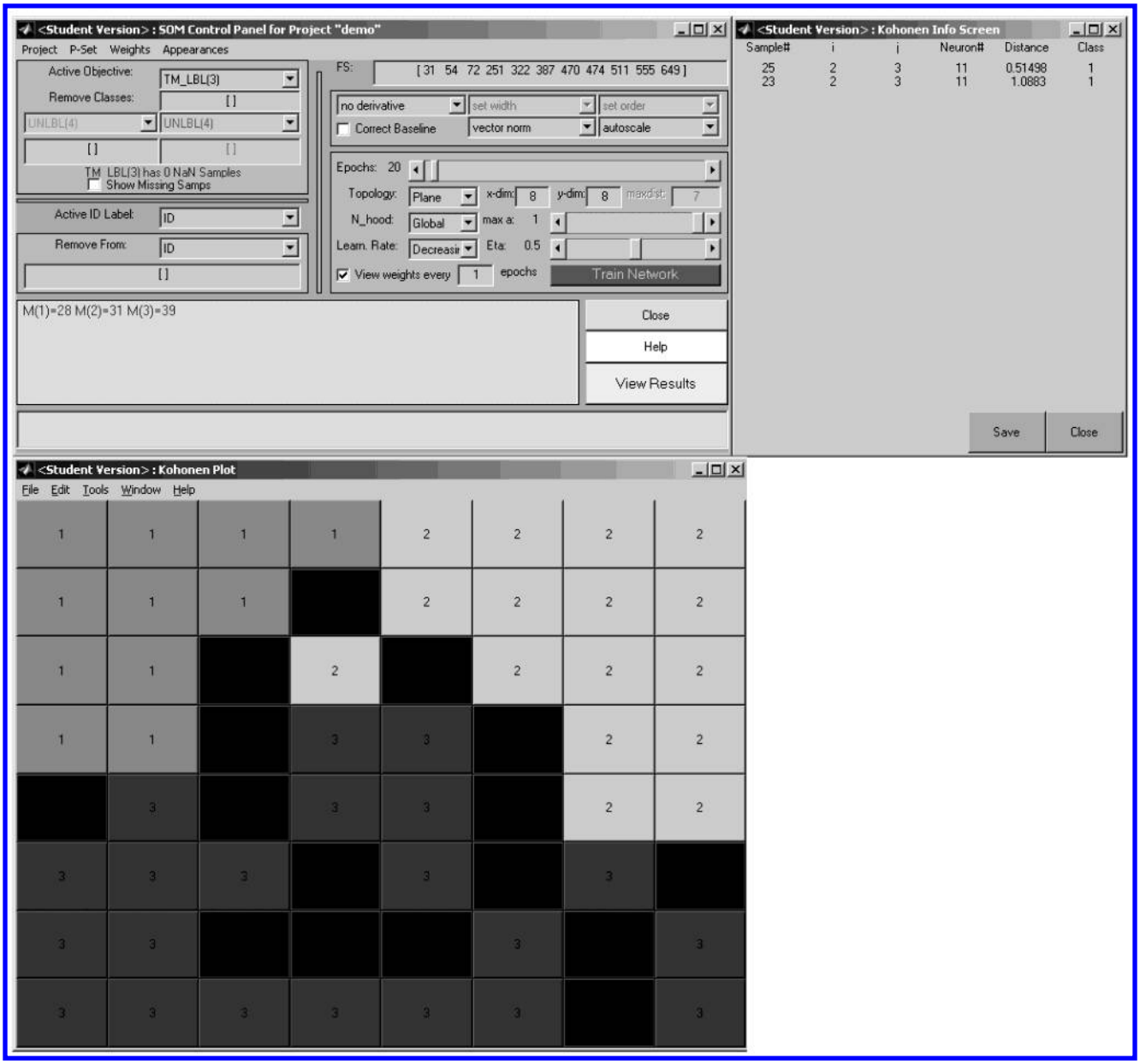

Figure 1. Onscreen layout of Somproj is shown. Control parameters include the number of epochs, grid topology and size, the neighborhood function, the neighborhood adjustment factor, the learning rate function, and the learning rate.

distributions with each distribution having the same covariance matrix but a different population mean.

Global ordering is usually achieved very quickly. However, occasional twists can distort the grid. This means that neighboring neurons in the grid do not reflect the same area of the data space. Points that excite adjacent neurons are actually dissimilar. However, such a twist is easily recognized (see Figure 2b). Figure 2c shows an arrangement of neurons that are not as easily identified as poor. Only after viewing the grid in relation to the data in Figure 2d does it become evident that distortion exists. In this case, the mapping error is actually worse than for the grid with the obvious twist (see Figure 2b), where each major cluster of the data is mapped reasonably well. It is only the orientation of the clusters with respect to each other that is incorrectly modeled. However, the distortion spans the two clusters in Figure 2d. Samples from one cluster appear to be similar to points from the other. Furthermore, points that are actually similar appear to be dissimilar on the map. This illustrates the importance of verifying the results of the neural network mapping.

Figure 2 was generated from a grid of neurons in a planar configuration. With a planar configuration, identifying kinks and twists is straightforward. However, each neuron is not equivalent. Neurons on the corners of the grid will have fewer direct neighbors than edge neurons, which will in turn have fewer direct neighbors than the remaining grid members. Therefore, a sample that excites an edge or corner neuron will not affect the adjustment of the weights to the same extent as will a sample exciting a neuron in the interior of the grid.

The remedy for this deficiency is to allow opposite edges of the grid to connect to each other. One can imagine the planar grid of neurons rolled into a cylinder. In this toroidal configuration, each neuron has the same neighborhood relationships as every other neuron. Therefore, each sample will have an equal influence on the development of the

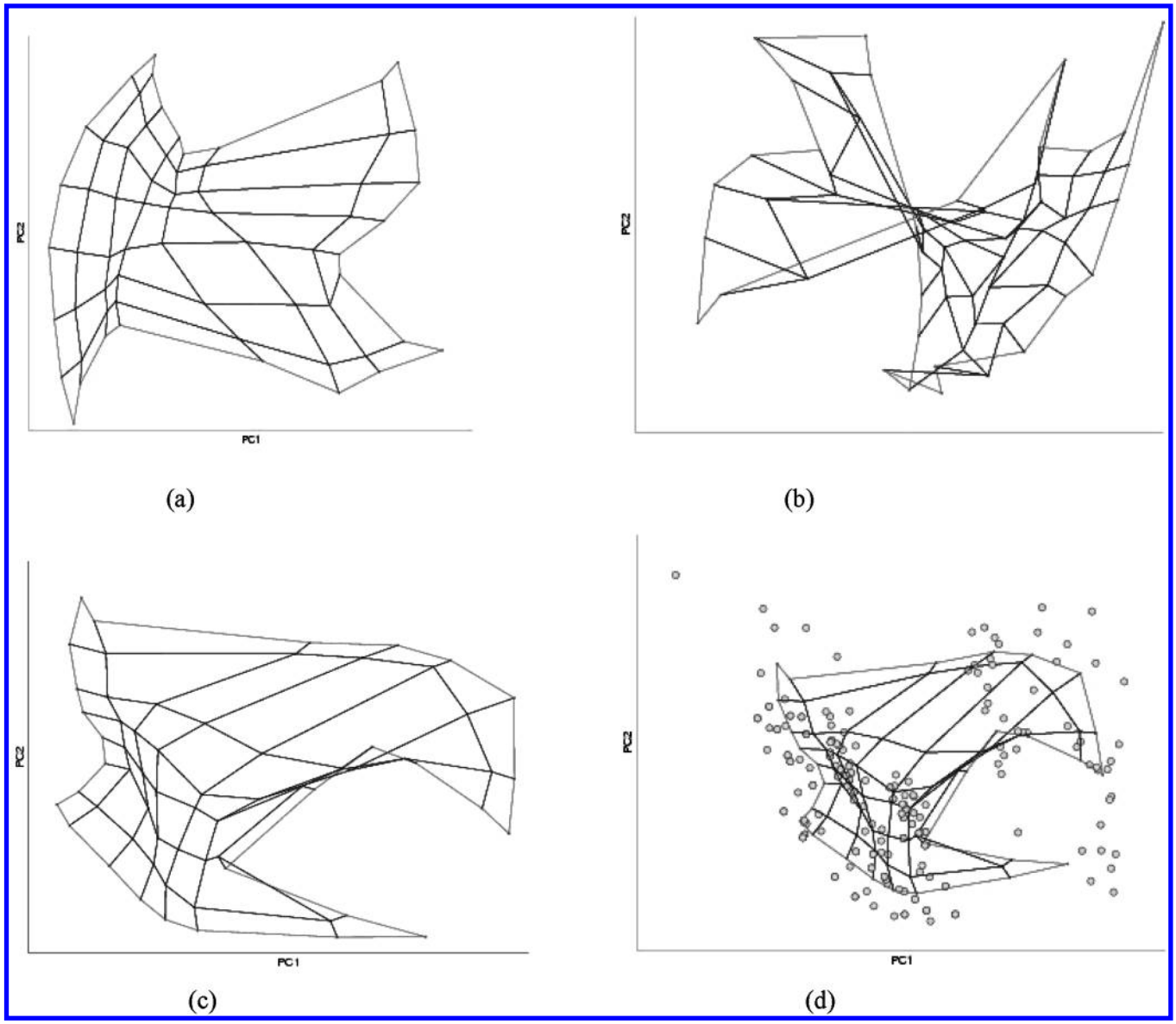

Figure 2. Visualization of neural network weights using the two largest principal components developed from the weights with the grid connections between weights shown: (a) a grid with good global ordering that will map the data well, (b) an arrangement with an obvious twist, (c) a grid with no obvious kinks, and (d) overlaying the data on top of the previous grid indicates it produces an inferior mapping of the data.

network. The disadvantage of this method is that it is much more difficult to determine the level of mapping error through the visualization techniques implemented in somproj.

Training. The single most important parameter to training a neural network is how the weights adjust when the signal is passed to it. For the Kohonen neural network, there is a set of interrelated parameters that control this process. First is the learning function, which controls how quickly the weights adjust to the signal. Two modes have been implemented: constant and decreasing. In the constant mode, η remains at the user specified value for the length of the training. In the decreasing mode, a linear function causes η to reduce over time. For a small number of epochs, the decreasing learning rate is more efficient than a constant one.

The neighborhood function controls the neurons that are adjusted to match the signal input. Initially, continuous, stepwise, and hybrid modes were implemented. All three are controlled by a maximum adjustment factor, *a*, which gives the adjustment for the winning neuron. In the continuous mode, an exponential function is fit to the distance from the winning neuron to its neighbors in order to determine their adjustment factors. This function decreases with time so that eventually only the winning neuron is adjusted. The stepwise mode adjusts all neighbors up to a maximum distance, *n*, to be adjusted at *a/2*. This mode is also a function of time. At each epoch the maximum distanced is reduced, until only the winning neuron is adjusted. The hybrid mode can be viewed as a combination of these two methods with the distance from the winning neuron adjusted using a linear function to determine neighborhood adjustment factors. After a distance of *n* steps, this adjustment is dropped to zero. The maximum distance is reduced during each epoch until only the winning neuron is adjusted. These functions are shown in Figure 3a−c.

It was determined that global ordering was not always a priority for these methods since the neighborhood function would drop off too quickly with time. This was true even when the corrections for the weights were allowed to last

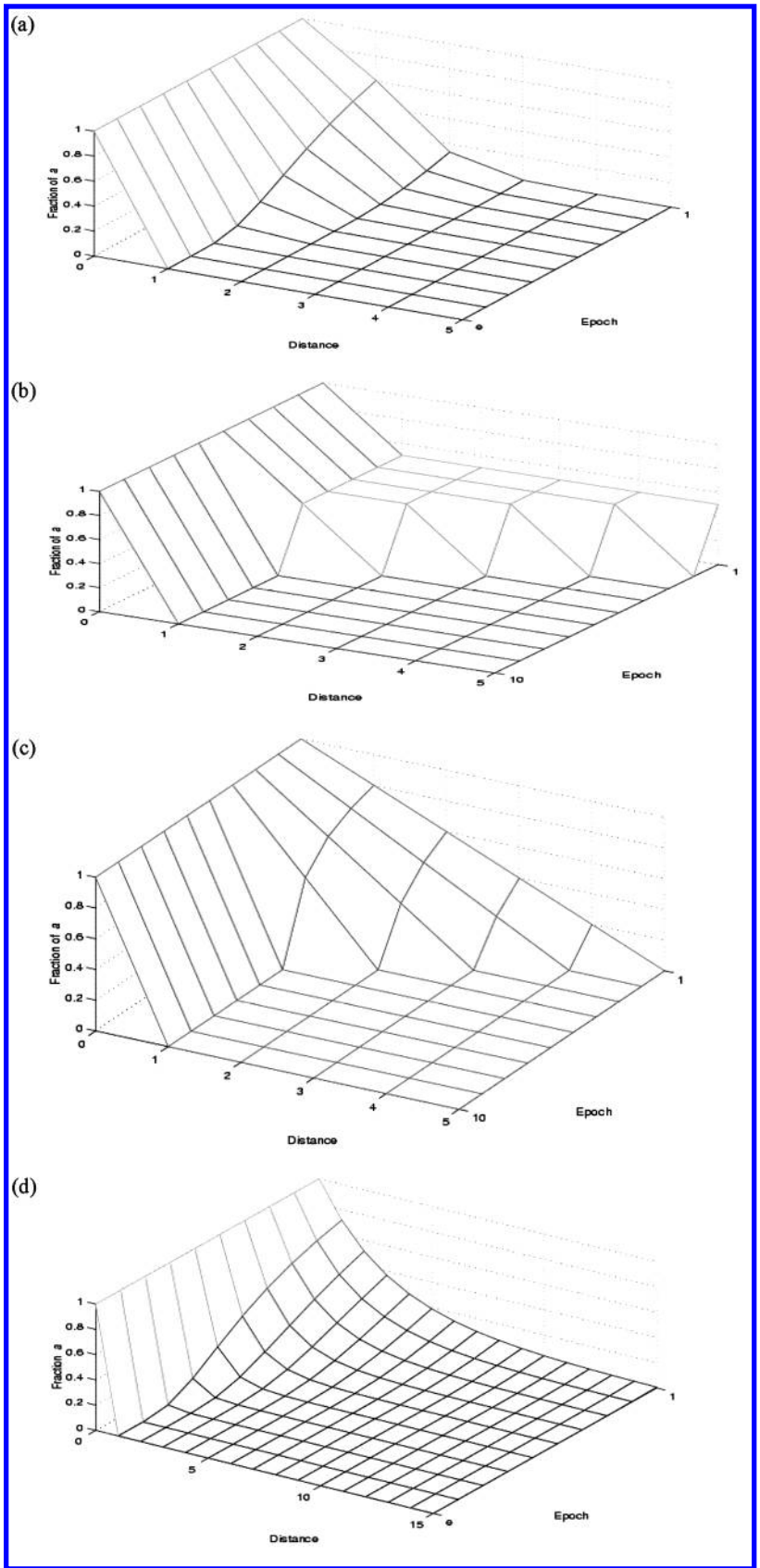

Figure 3. SOM neighborhood functions are shown. (a) The continuous neighborhood function drops off with time from the first epoch to the last epoch. (b) The stepwise function drops off at a rate not determined by the number of epochs but by *n*. (c) The hybrid neighborhood function drops off at a rate also determined by *n*. (d) The global neighborhood function shown is a modification of the continuous function.

within the same neighborhood for a multiple number of epochs. Therefore, a "global" neighborhood function was developed (see Figure 3d). The function is a modification of the continuous function and involves increasing the width of the exponential function and decreasing the rate at which it drops off with time.

Feature Selection. It is sometimes necessary to perform variable selection before generating a Kohonen neural

network map of the data. Consider a data set consisting of 30 samples distributed between 3 classes (bad, good, and better). Each sample is characterized by 30 measurements. Suppose that only 4 of these measurements contain information about the classification problem of interest. A Kohonen neural network map developed from these 30 measurements may not show the desired clustering, whereas a self-organizing map developed from the four informative measurements will exhibit clustering on the basis of class.

If each data point is tagged with a class label, a genetic algorithm can be used to select the discriminating features for mapping. Each SOM generated for each feature subset extracted from its chromosome is scored for clustering by eq 2 where K_N is the number of nearest samples with the same class label, SHC is the number of K_N nearest neighbors with the same class label found in the selected and neighboring neurons as the sample in question, and SW(s) is the sample weight. K_N is a user specified parameter and is often set equal to the number of samples in each class. The only deficiency of this scoring scheme is that neurons in the center of a widely dispersed class can be underscored and neurons that are adjacent to other neurons containing samples from other classes may be over scored.

$$F = \sum_{N}\sum_{S\in N} \frac{1}{K_N(s)} \times \mathrm{SHC(s)} \times \mathrm{SW(s)} \tag{2}$$

To facilitate the tracking and scoring of the SOM, class and sample weights, which are an integral part of the fitness function, are computed (see eqs 3 and 4) where CW(c) is the weight of class c (with c varying from 1 to the total number of classes in the data set). $SW_c(s)$ is the weight of sample s in class c. Class weights sum to 100, and sample weights for the objects comprising a particular class sum to a value equal to the class weight of the class in question.

$$\mathrm{CW}(c) = 100\frac{\mathrm{CW}(c)}{\sum_{c}\mathrm{CW}(c)} \tag{3}$$

$$\mathrm{SW}(s) = \mathrm{CW}(c)\frac{\mathrm{SW}(s)}{\sum_{s\in c}\mathrm{SW}(s)} \tag{4}$$

The fitness function of the GA is able to focus on those samples and classes that are difficult to classify by boosting their weights over successive generations. Sample weights are adjusted individually, but the adjustment is based on the neuron closest to the sample. If three samples are in the same neuron, they will be adjusted by the same amount.

To boost, it is necessary to compute both the sample-hit rate (SHR), which is the mean value of SHC/K_c over all feature subsets produced in a particular generation (see eq 5), and the class-hit rate (CHR), which is the mean sample hit rate of all samples in a class (see eq 6). ϕ in eq 5 is the number of chromosomes in the population, and AVG in eq 6 refers to the average or mean value. During each generation, class and sample weights are adjusted by a perceptron (see eqs 7 and 8) with the momentum, P, set by the user. ($g + 1$ refers to the current generation, whereas g is the previous generation.) Classes with a lower class hit rate and samples with a lower sample hit rate are boosted more heavily than those classes or samples that score well.

$$\mathrm{SHR}(s) = \frac{1}{\phi}\sum_{i=1}^{\phi}\frac{\mathrm{SHC}_i(s)}{K_c} \tag{5}$$

$$\mathrm{CHR}_g(c) = \mathrm{AVG}(\mathrm{SHR}_g(s):\forall_{\mathrm{sec}}) \tag{6}$$

$$\mathrm{CW}_{g+1}(s) = \mathrm{CW}_g(s) + P(1 - \mathrm{CHR}_g(s)) \tag{7}$$

$$\mathrm{SW}_{g+1}(s) = \mathrm{SW}_g(s) + P(1 - \mathrm{SHR}_g(s)) \tag{8}$$

The change in the class weights is monitored throughout the run. If the average change in the class weights is greater than some tolerance, the genetic algorithm is said to be learning its optimal class weights. Once this tolerance has been reached, the class weights become fixed, and the sample weights in each class become uniformly distributed according to their class weight. This initiates the second stage. The momentum, which controls the rate at which the sample weights are changed, is initially assigned a value of 0.8, while the genetic algorithm is learning, but the momentum is adjusted to 0.4 once the class weights become fixed. These values have been chosen in part because they facilitate learning by the genetic algorithm but do not cause a particular sample or class to dominate the calculation, which would result in the other samples or classes not contributing to the scoring by the fitness function.

During each generation, class and sample weights are updated using class and sample hit-rates from the previous generation. Potential solutions (i.e. feature subsets) are evaluated, and their SOMs are scored by eq 2. Solutions with a higher fitness score will have a higher probability of being selected for recombination, which involves a structured yet randomized exchange of information with the possibility that good solutions can generate even better ones. Additional variability within the population of solutions is achieved using the mutation operator. The boosting algorithm adjusts the internal parameters of the GA for the next iteration. Evaluation, reproduction, and boosting of potential solutions are repeated until a specified number of generations are executed or a feasible solution (i.e., a high fitness score for a particular feature subset in the population) is found.

RESULTS AND DISCUSSION

Plastics. The first data set[15,16] used to compare principal component maps with SOMs consisted of 188 Raman spectra of six common household plastics: high-density polyethylene (HDPE), low-density polyethylene (LDPE), poly(ethylene terephthalate) (PET), polypropylene (PP), polystyrene (PS), and poly(vinyl chloride) (PVC). The overall goal of the study was to differentiate common household plastics by type for recycling purposes using Raman spectroscopy. Sorting of plastics by type is crucial for recycling because the most valuable reprocessed plastics are prepared from pure polymer streams.

The 188 plastic containers used in this study were collected from residential homes and BFI Recycling in Pocatello, ID. Each plastic sample was cut from collected containers. A Spex 500M 1/2 meter Raman spectrometer, which incorpo-

Table 1. Training Set

number of spectra	plastic type
33	high-density polyethylene (HDPE)
26	low-density polyethylene (LDPE)
35	polyethylene terphthalate (PET)
26	polypropylene (PP)
32	polystyrene (PS)
17	poly(vinyl chloride) (PVC)
169	total

Table 2. Prediction Set

number of spectra	plastic type
5	high-density polyethylene (HDPE)
2	low-density polyethylene (LDPE)
5	polyethylene terphthalate (PET)
2	polypropylene (PP)
5	polystyrene (PS)
0	poly(vinyl chloride) (PVC)
19	total

rated Spex Model 1489 collection optics module, an Omnichrome Model 160 T/B air-cooled Ar+ laser, and a liquid nitrogen cooled charged coupled detector device was used to obtain spectra of the cut plastic. The sample geometry was chosen based on optimal placement in the sample holder of the spectrometer.

Each Raman spectrum, which was an average of 16 one-second scans, collected over the wavenumber range 850−1800 cm^{-1} for 1093 points was boxcar averaged every 10 points to yield 218-point spectra, baseline corrected for offsets using a linear polynomial, and then normalized to unit length to adjust for variations in the optical path length. The 188 spectra were divided into a training set of 169 spectra (see Table 1) and a prediction or validation set of 19 spectra (see Table 2). Spectra in the training and prediction sets were chosen by random lot. For pattern recognition analysis, each sample or Raman spectrum was represented by a data vector, $x = (x_1, x_2, x_3 \ldots x_j, x_{218})$ where x_j is the Raman intensity of the jth point from the baseline corrected normalized Raman spectrum.

The first step in the study was to apply principal component analysis to the data. Each spectrum was normalized to unit length and mean centered prior to principal component analysis. Figure 4 shows a plot of the two largest principal components of the 218-point Raman spectra comprising the training set. Each Raman spectrum is represented as a point in the principal component plot (1 = HDPE, 2 = LDPE, 3 = PET, 4 = PP, 5 = PS, and 6 = PVC). Clustering of spectra by plastic type in the plot is evident. When the prediction set samples were projected onto the principal component map, 17 of the 19 samples were found to lie in a region of the map with other samples that had the same class label (see Figure 5). One HDPE sample was misclassified as LDPE, and one LDPE sample was misclassified as HDPE. This was not surprising in view of the overlap between these two classes (HDPE and LDPE) in the map due to the similarity of their spectra.

The next step was to use a SOM to map the 218-dimensional spectral data. For the plastic data set, the learning rate of the Kohonen neural network was varied from 0.05 to 0.001 during training. Each spectrum in the training set was trained 50 times. During training, the development of the weights was monitored. A principal component plot of the weights, showing the grid connection between neurons, was displayed every epoch. No twists on the grid were observed during the final phase of the training.

The number of neurons used to study the training set data was 64 (i.e., an 8 × 8 grid in a toroidal configuration). Selection of the map's grid size is crucial to ensure an accurate mapping of the data. Based on our previous experience, the number of neurons used should be between 33% and 50% of the number of spectra in the training set.

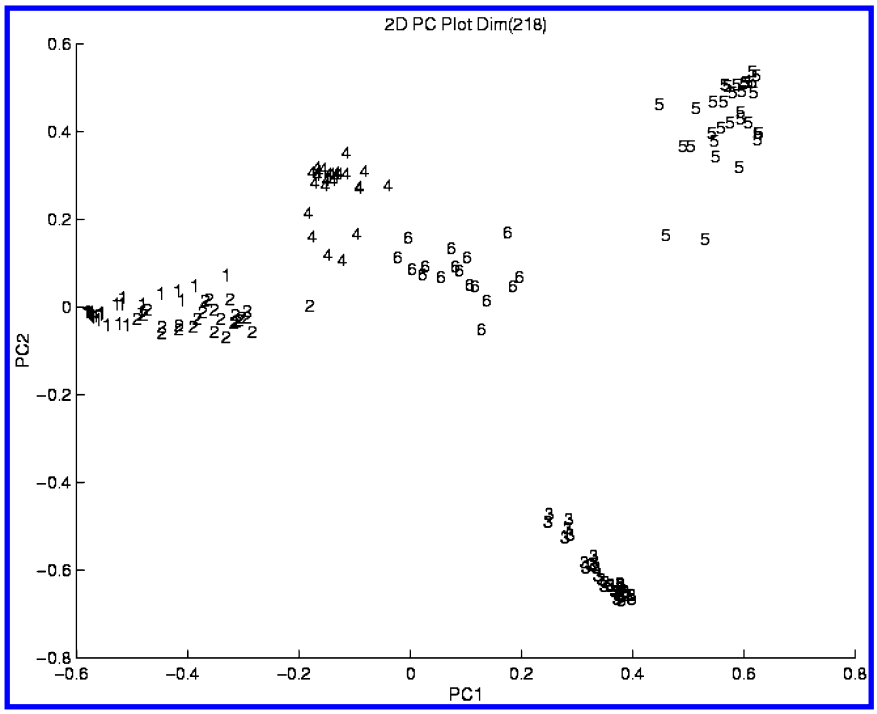

Figure 4. A plot of the two largest principal components of the normalized and mean centered 218-point spectra that comprise the training set. Each spectrum is represented as a point in the principal component map: 1 = HDPE, 2 = LDPE, 3 = PET, 4 = PP, 5 = PS, and 6 = PVC.

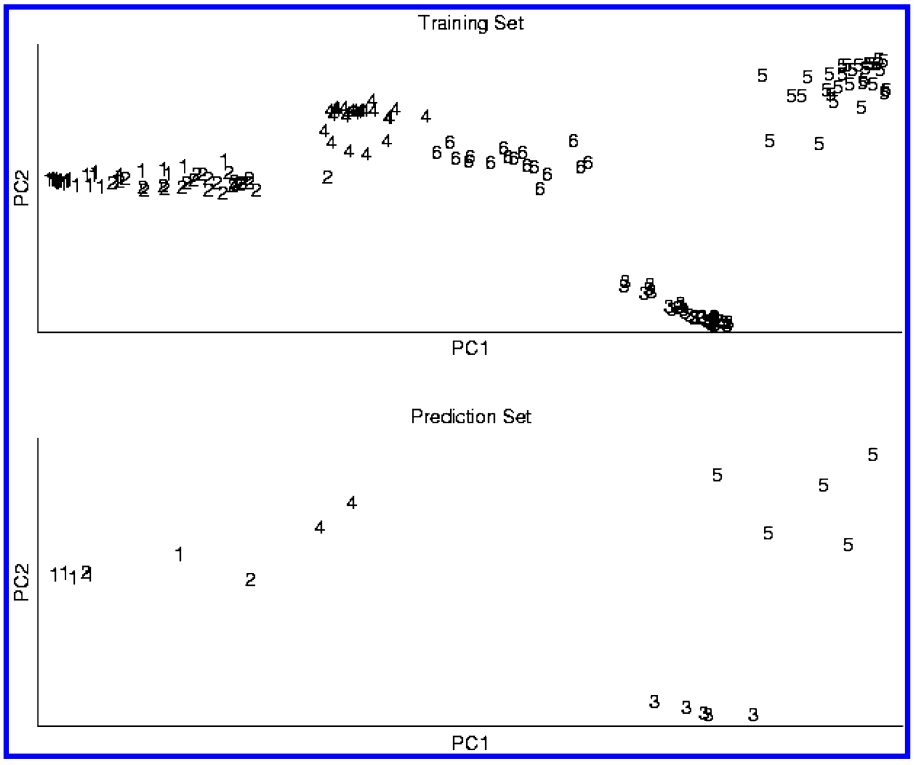

Figure 5. Prediction set samples projected onto the principal component map developed from the 218-point spectra comprising the training set. Each spectrum is represented as a point in the map: 1 = HDPE, 2 = LDPE, 3 = PET, 4 = PP, and 5 = PS.

Too few neurons produces a map of the data that does not faithfully reflect the structure of the high-dimensional measurement space, whereas too many neurons generates a map of the data that is not aesthetically pleasing since there will be many neurons that do not respond to any of the spectra making it more difficult to discern clustering. These conclusions admittedly depend on the data itself with different workers having different ideas on what are the best parameters.

After the weights for the neurons were trained, distances were calculated between each sample and the neurons with each samples assigned to the nearest neuron. Each neuron was given a class label based on the class assignment of the samples responding to it. In all cases, only samples with the same class label responded to a given neuron. Figure 6 shows a self-organizing map generated from the training set data. Each point on the map was entered as a 1 (HDPE), 2 (LDPE), 3 (PET), 4 (PP), 5 (PS), or 6 (PVC). The self-organizing map developed from the training set data consisted of 169 Raman spectra and 218 spectral variables. The large centered numbers in Figure 6 represent the class assignments of the 169 training set samples. On the basis of these assignments, one can observe clustering of the spectra by plastic type.

Prediction set samples were then passed to the network and assigned to the nearest neuron. They are represented as superscripts in Figure 6. Each prediction set sample is represented as a 1 (HDPE), 2 (LDPE), 3 (PET), 4 (PP), or 5 (PS).

All 19 prediction set samples were assigned to neurons that had spectra with the same class label. The validation of the self-organizing maps using these 19 prediction set samples implies that classification of the plastics by the SOMs was as reliable as that obtained by other methods and more importantly the classification was obtained without any assumptions about the data.

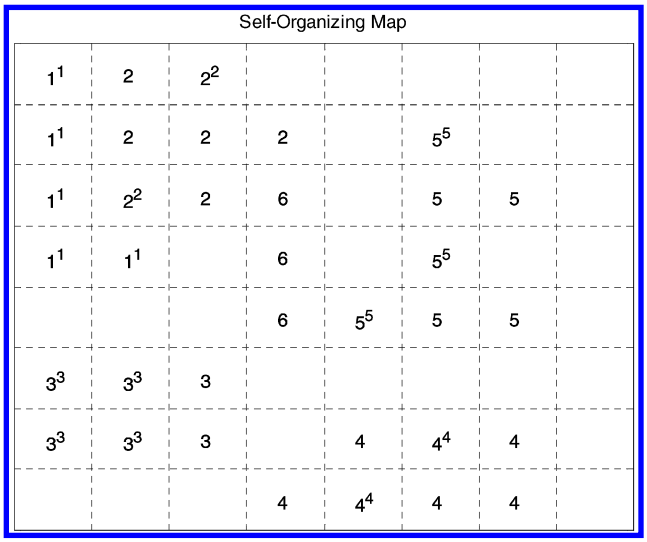

Figure 6. Self-organizing map generated from the training set data with the prediction set samples projected onto the map. Each point from the training and prediction sets was entered on the map as 1 (HDPE), 2 (LDPE), 3 (PET), 4 (PP), 5 (PS), or 6 (PVC).

The data used to train the Kohonen neural network was only normalized to unit length. Mean centering would have probably produced the same result. As for autoscaling, the amount of noise that would have been introduced by amplifying the baseline would probably have proven deleterious.

Testing of Raw Materials. The second data set[17,18] consisted of acceptable and rejected samples of PH102 avicel. The goal of the study was to develop a method to differentiate acceptable from unacceptable lots of PH102 avicel using near-infrared (NIR) spectroscopy. The training set consisted of 34 acceptable samples and 30 rejected samples, whereas the prediction set contained 14 acceptable samples.

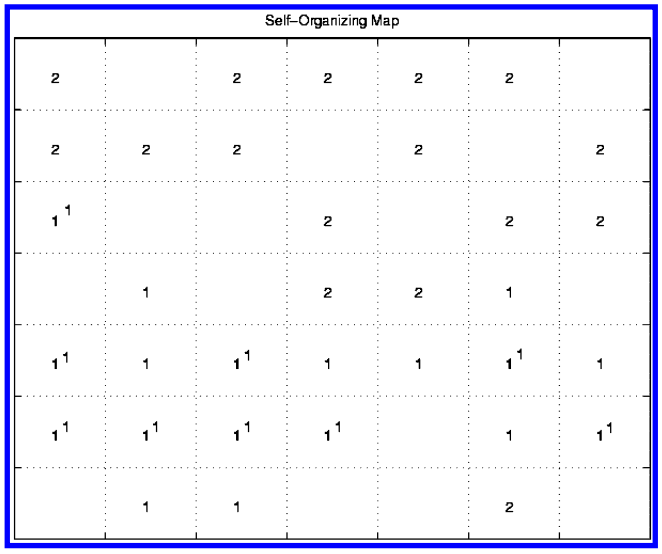

Figure 7. Self-organizing map generated from the training set data with the prediction set samples projected onto the map. Each point from the training and prediction sets was entered on the map as 1 (acceptable) or 2 (unacceptable).

Spectral data were collected using with a Technicon InfraAlyzer 500S NIR spectrophotometer. The NIR spectra were recorded from 1100 to 2500 nm at 4 nm intervals. The samples were placed in a standard sample cup supplied with the instrument and measured against a highly reflective ceramic disk, which served as the reference.

Each raw spectrum was normalized to unit length to adjust for variations in the optical path length. For pattern recognition analysis, each avicel spectrum was represented as a data vector $x = (x_1, x_2, x_3...x_j...x_{350})$ where x_j is the absorbance of the jth point. Both principal component analysis and the Kohonen neural network were first applied to mapping the training set data. However, clustering of the spectra by type (acceptable versus unacceptable) did not occur in either the principal component plot or SOM developed from the 350-point spectra. Furthermore, one of the raw training set spectra (an unacceptable avicel sample) was as an outlier in both the principal component plot and SOM and was therefore deleted from the analysis.

Our previous experience has shown that all pattern recognition methods will work well when the problem is simple. By identifying the appropriate features, a "hard" problem can be reduced to a "simple" one. Therefore, a genetic algorithm was used to identify discriminating wavelengths in the raw spectra. The pattern recognition GA identified features by sampling key feature subsets, scoring their SOMs, and tracking those classes and/or samples that were most difficult to classify. The boosting routine used this information to steer the population to an optimal solution. After 50 generations, the pattern recognition GA identified 8 wavelengths whose SOM showed clustering of the near-infrared spectra on the basis of raw material quality (see Figure 7), which suggests that information is contained in the NIR spectra of avicel characteristic of its quality.

A validation set of 14 near-infrared spectra was used to assess the predictive ability of the 8 spectral features identified by the pattern recognition GA. The prediction set samples were projected onto the SOM developed from the 63 training set samples and 8 spectral features identified by the pattern recognition GA. Figure 7 shows the projection of the prediction set samples onto the SOM defined by the 8 spectral features. The large centered numbers in Figure 7 represent the class assignments of the training set samples, and the superscripted numbers represent the class assignments of the prediction set samples. Each prediction set sample is assigned to a neuron occupied by avicel samples that have the same class label. Evidently, information about product quality can be obtained from raw near-infrared spectra with minimal data preprocessing when SOMs are used to analyze the data.

CONCLUSION

Principal component analysis and Kohonen self-organizing maps allow multivariate data to be displayed as a graphfor direct viewing thereby extending the ability of human pattern recognition to uncover obscure relationships in complex data sets. A major advantage of the Kohonen neural network over principal component analysis is that data preprocessing is minimal. Both principal component analysis and the Kohonen neural network enable a scientist or engineer to play a more interactive role in the data analysis. Clearly, these two techniques can be very useful when an investigator believes that distinct class differences exist in a collection of samples but is not sure about the nature of the classes.

REFERENCES AND NOTES

(1) Jolliffe, I. T. *Principal Component Analysis*; Springer-Verlag: 1986.
(2) Jackson, J. E. *A User's Guide to Principal Components*; John Wiley & Sons: 1991.
(3) Wold, S.; Esbenson, K.; Geladi, P. Principal Component Analysis. *Chem. Intel. Lab. Systems* **1987**, *2*, 37−52.
(4) Smeyers-Verbeke, J.; Den Hartog, J. C.; Dekker: W. H.; Coomans, D.; Buydens, L.; Massart, L. *Atmos. Environ.* **1984**, *18*, 2741.
(5) Kowalski, B. R. Measurement Analysis by Pattern Recognition. *Anal. Chem*. **1975**, *47*, 1162A.
(6) Eide, M. O.; Kvalheim, O. M.; Telnaes, N. Routine analyses of Crude Oil Fractions by Principal Component Modeling of Gas Chromatographic Profiles. *Anal. Chim. Acta* **1986**, *191*, 433−437.
(7) Kvalheim, O. M. Oil−−Source Correlation by the Combined use of Principal Component Modeling, Analysis of Variance and a Coefficient of Congruence. *Chem. Intel. Lab. Systems* **1987**, *2*, 127−136.
(8) Grahn, H.; Delaglio, F.; Delsuc, M. A.; Levy, G. C. Multivariate Data Analysis for Pattern Recognition in Two-Dimensional NMR. *J. Magn. Reson*. **1988**, *77*, 294−307.
(9) Espen, P. V.; Adams, F. The Application of Principal Component and Factor Analysis Procedures to Data for Element Concentrations in Aerosols from a Remote Region. *Anal. Chim. Acta* **1983**, *150*, 153−161.
(10) Kowalski, B. R.; Bender, C. F. Pattern Recognition. II. Linear and Nonlinear Methods For Displaying Chemical Data. *J. Am. Chem. Soc*. **1973**, *95*, 686−693.
(11) Kohonen, T. *Self-Organizing Maps*, 3rd ed.; Springer-Verlag: 2000.
(12) Zupan, J.; Gasteiger, J. *Neural Networks for Chemists*; VCH Publishers: 1993.
(13) Davidson, C. E. Genetic Algorithms for Data Mining and Multivariate Data Analysis, Ph.D. Thesis, Clarkson University, Potsdam, NY, 2003.
(14) Lavine, B. K.; Davidson, C. E. Genetic Algorithms for Data Mining − Profiting from the Past. *J. Chemom*. **2004**, submitted for publication.
(15) Allen, V.; Kalivas, J. H.; Rodriguez, R. G. Post-Consumer Plastic Identification Using Raman Spectroscopy. *Appl. Spectrosc*. **1999**, *53*(6), 672.
(16) Lavine, B. K.; Moores, A. J. In *Pattern Recognition, Chemometrics, and Imaging for Optical Environmental Monitoring*; Siddiqui, K., Eastwood, D., Eds.; *Proceedings of SPIES*, 1999; pp 103−112.
(17) Shah, N. K.; Gemperline, P. J. Combination of the Mahalanobis Distance and Residual Variance Pattern Recognition Techniques for Classification of Near-infrared Reflectance Spectra. *Anal. Chem*. **1990**, *62*, 465.
(18) Lavine, B. K.; Davidson, C. E.; Moores, A. J. Genetic Algorithms for Spectral Pattern Recognition. *Vib. Spectrosc*. **2002**, *28*(1), 83.

CI030039Y