REFERENCES

[1] R. Adams, "Radial Decomposition of Discs and Spheres," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, vol. 55, no. 5, pp. 325–332, Sept. 1993.

[2] J. Bresenham. "Algorithm for Computer Control of Digital Plotter, *IBM System J.*, vol. 4, pp. 25–30, 1965.

[3] J. Bresenham. "A Linear Algorithm for Incremental Display of Circular Arcs," *Comm. ACM*, vol. 20, no. 2, pp. 100–106, 1977.

[4] B. Chaudhuri. "An Efficient Algorithm for Running Window Pel Gray Level Ranking in 2D Images," *Pattern Recognition Letters*, vol. 11, no. 2, pp. 77–80, 1990.

[5] S. Crabtree, L.-P. Yuan, and R. Ehrlich, "A Fast and Accurate Erosion-Dilation Method Suitable for Microcomputers," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, vol. 53, no. 3, pp. 283–290, May 1991.

[6] M. Van Droogenbroeck, "On the Implementation of Morphological Operations," *Math. Morphology and Its Applications to Image Processing*, J. Serra and P. Soille, eds. Dordrecht: Kluwer Academic Publishers, 1994, pp. 241–248.

[7] R. Haralick, S. Sternberg, and X. Zhuang, "Image Analysis Using Mathematical Morphology," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 4, pp. 532–550, July 1987.

[8] B. Laÿ, "Recursive Algorithms in Mathematical Morphology," *ACM Trans. Graphics*, vol. 6, pp. 691–696, 1987.

[9] J. Pecht, "Speeding Up Successive Minkowski Operations," *Pattern Recognition Letters*, vol. 3, no. 2, pp. 113–117, 1985.

[10] I. Ragnemalm, "Fast Erosion and Dilation by Contour Processing and Thresholding of Distance Maps," *Pattern Recognition Letters*, vol. 13, pp. 161–166, 1992.

[11] J. Serra, *Image Analysis and Mathematical Morphology*. London: Academic Press, 1982.

[12] J. Serra, "Examples of Structuring Functions and Their Uses," *Image Analysis and Mathematical Morphology, Volume 2: Theoretical Advances*, J. Serra, ed., ch. 4, pp. 71–99. Academic Press, 1988.

[13] J. Serra and L. Vincent, "An Overview of Morphological Filtering," *Circuits Systems Signal Process*, vol. 11, no. 1, pp. 47–108, 1992.

[14] R. van den Boomgaard and R. van Balen, "Methods for Fast Morphological Image Transforms Using Bitmapped Binary Images. *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, vol. 54, no. 3, pp. 252–258, May 1992.

[15] M. van Herk, "A Fast Algorithm for Local Minimum and Maximum Filters on Rectangular and Octogonal Kernels, *Pattern Recognition Letters*, vol. 13, pp. 517–521, 1992.

[16] L. Vincent, "Morphological Transformations of Binary Images with Arbitrary Structuring Elements," *Signal Processing*, vol. 22, no. 1, pp. 3–23, Jan. 1991.

[17] I. Young, R. Pevereni, P. Verbeek, and P. Otterloo, "A New Implementation for the Binary and Minkowski Operators," *Computer Graphics and Image Processing*, vol. 17, pp. 189–210, 1981.

Weighted Parzen Windows for Pattern Classification

Gregory A. Babich and Octavia I. Camps

Abstract—This correspondence introduces the *weighted-Parzen-window* classifier. The proposed technique uses a clustering procedure to find a set of reference vectors and weights which are used to approximate the *Parzen-window* (*kernel-estimator*) classifier. The weighted-Parzen-window classifier requires less computation and storage than the full Parzen-window classifier. Experimental results showed that significant savings could be achieved with only minimal, if any, error rate degradation for synthetic and real data sets.

Index Terms—Nonparametric classifiers, Parzen-windows, kernel estimator, clustering, training samples, discriminant analysis, Bayes error, leave-one-out, holdout.

1 INTRODUCTION

WHEN designing a pattern recognition system, *nonparametric classifiers* are often used. Nonparametric techniques do not assume a particular form of density function but usually estimate one. Since the underlying density of real data rarely fits common density models [2], the nonparametric classification approach is a good choice for practical applications. The well-known *Parzen-window* (*kernel-estimator*) classifier is a popular nonparametric approach because of its excellent performance and firm theoretical foundation. Unfortunately, the Parzen-window (PW) approach can require significant computational resources in terms of processing time and storage.

Researchers have developed less costly techniques to implement PW-type classifiers. One approach is to find a subset of a set of *training samples* to estimate the PW density. Fukunaga [5] discusses a procedure in which a nonparametric data reduction technique is used to split training data into two separate sets by an iterative swapping scheme that is controlled by an entropy criterion function. By splitting the data, a suitable subset of the training data may be found and subsequently used for PW classification. This approach is referred to as the *reduced Parzen* (RP) *classifier* [5]. In the case of the RP classifier, the final split is generally dependent on the initial partition [5]. Another approach is to reduce the size of the training set by *clustering*. One such procedure, proposed by West [8], is called *collapsing mixtures*. In this approach, the training data are clustered to find centers and weights which are used in a kernel estimator. West suggests monitoring plots of the marginal distributions during the clustering process so that training may be terminated when significant changes are detected [8]. Other parsimonious approaches are the *binned kernel estimators*. The traditional binned estimator requires a grid of equally spaced bin centers, which are used in conjunction with available training data to estimate its density function. The reader is referred to Fan and Marron [4] for a discussion of binned estimators.

• *G.A. Babich is with the Applied Research Laboratory, The Pennsylvania State University, PO Box 30, State College, PA 16804.*
E-mail: gab@arlvax.arl.psu.edu.
• *O.I. Camps is with the Department of Electrical Engineering, Department of Computer Science and Engineering, The Pennsylvania State University, 121 Electrical Engineering East, University Park, PA 16802.*
E-mail: camps@cse.psu.edu

Manuscript received Nov. 10, 1994; revised Oct. 4, 1995. Recommended for acceptance by D.M. Titterington.
For information on obtaining reprints of this article, please send e-mail to: transactions@computer.org, and reference IEEECS Log Number P95170.

In this correspondence, we introduce the *weighted-Parzen-window* (WPW) classification approach. This technique uses a clustering procedure to find a set of reference vectors and weights which are used to approximate the PW classifier. The training algorithm is discussed in Section 2. The WPW technique is most closely related to the collapsing mixture estimator and the RP classifier. However, the WPW approach differs from the collapsing mixture approach in that WPW training uses the PW estimate as a baseline during the training phase, which is terminated when the *distance* between the two estimates exceeds a threshold. In Section 3, experimental results are presented. Synthetic and real data sets are used to compare the WPW, PW, and RP classifiers. Section 4 gives concluding remarks.

2 Weighted Parzen Windows

Classification is often accomplished by *discriminant analysis*, where separate discriminant functions are used for each class of data. In this approach, unlabeled samples are given the class label of the largest valued discriminant function [2]. In the present correspondence, each class's discriminant function is the estimated density function for that class weighted by the corresponding a priori probability. Therefore, without loss of generality, the following discussion treats the case of a single class.

Given a set of n d-dimensional training samples $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, the Parzen-window (PW) density estimate is given by [2], [5], [7],

$$p_n(\mathbf{x}) = \sum_{k=1}^{n} \frac{1}{nh^d} \varphi\left(\frac{\mathbf{x}-\mathbf{x}_k}{h}\right),$$

where $\varphi(\cdot)$ is the *window* function and h is the window *width* parameter. Parzen showed that $p_n(\mathbf{x})$ converges to the true density if $\varphi(\cdot)$ and h are properly selected, and the distribution of $\mathbf{x}$ is continuous [2], [5], [7]. The window function is required to be a finite-valued non-negative density function where

$$\int \varphi(\mathbf{y})d\mathbf{y} = 1,$$

and the width parameter is required to be a function of n such that

$$\lim_{n\to\infty} h(n) = 0,$$

and

$$\lim_{n\to\infty} nh^d(n) = \infty.$$

A more general form of $p_n(\mathbf{x})$ is given by

$$p_n(\mathbf{x}) = \sum_{k=1}^{n} \frac{1}{n|H|} \varphi\left[H^{-1}(\mathbf{x}-\mathbf{x}_k)\right], \tag{2.1}$$

where H is a d-by-d diagonal matrix of width parameters and $|\cdot|$ is the determinant. This approach allows for smoothing in each of the d directions. Other forms of H are possible and are mentioned below.

The weighted-Parzen-window (WPW) technique approximates (2.1) using fewer samples. A clustering procedure is used to find a set of *reference* vectors (cluster centers) and window *weights*. Given a set of reference vectors $R = \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_m\}$, $m \le n$, the WPW approximation to (2.1) is given by

$$p_m(\mathbf{x}) = \sum_{k=1}^{m} \frac{w_k}{n|H|} \varphi\left[H^{-1}(\mathbf{x}-\mathbf{r}_k)\right], \tag{2.2}$$

where w_k is the kth window weight and is equal to the number of original training samples that were collapsed into $\mathbf{r}_k$.

Fig. 1 gives a specific training algorithm, although many variations are possible. The WPW training algorithm finds a set of reference vectors by combining the two samples that are closest to each other for each training step. Whenever two samples are combined in this manner, they forever become members of the same cluster; this is known as *hierarchical clustering* [2]. Note that the error measure of the training algorithm approximates the L_1 distance given by $\int |p_n - p_m| d\mathbf{x}$, whose range is [0, 2] which is therefore the range of e.

```
1) Choose: $H$, and $e_{max} \ge 0$.
2) Initialize: $m \leftarrow n$, $R \leftarrow X$, $\mathbf{w} = [w_1\ w_2 \ldots w_m]^T = \mathbf{1}$.
3) Calculate and store $p_n(\mathbf{x}_k)$, $k = 1, \ldots, n$, using (2.1). Initialize: $p_m(\mathbf{x}_k) \leftarrow p_n(\mathbf{x}_k)$.
4) Choose two vectors which minimize the distance $\|H^{-1}(\mathbf{r}_i - \mathbf{r}_j)\|$, $i \ne j$.
5) Calculate the vector $\mathbf{r}_o = \frac{w_i\mathbf{r}_i + w_j\mathbf{r}_j}{w_i + w_j}$.
6) Update $R$ such that $\{\mathbf{r}_i, \mathbf{r}_j\} \notin R$ and $\mathbf{r}_o \in R$, $w_o \leftarrow w_i + w_j$, $m \leftarrow m - 1$.
7) Update $p_m(\mathbf{x}_k) \leftarrow p_m(\mathbf{x}_k) + \frac{1}{n|H|}\{w_o\varphi[H^{-1}(\mathbf{x}_k - \mathbf{r}_o)] - w_i\varphi[H^{-1}(\mathbf{x}_k - \mathbf{r}_i)] - w_j\varphi[H^{-1}(\mathbf{x}_k - \mathbf{r}_j)]\}$.
8) Calculate $e = \frac{1}{n}\sum_{k=1}^{n}\frac{|p_n(\mathbf{x}_k) - p_m(\mathbf{x}_k)|}{p_n(\mathbf{x}_k)}$.
9) If $[(e < e_{max})$ AND $(1 < m)]$, go to 4.
10) If $(e > e_{max})$ replace $\mathbf{r}_i$, $\mathbf{r}_j$, $w_i$, $w_j$. Output $R$ and $\mathbf{w}$.
```

Fig. 1. WPW training algorithm for a single class of data.

The WPW approach has several attractive qualities.

1) The WPW training algorithm makes use of *agglomerative* hierarchical clustering [1], [2]. This form of clustering has been well studied, and efficient algorithms exist for its implementation [1]. Note that the error function of Step 8 is recursively updated and is therefore efficient as well.
2) The training procedure is completely unique. This property can be used in the design process as follows. The training set can be reduced to a single cluster while recording the cluster indices and the value of e at each training step. Subsequently, a plot of e as a function of the level of training (reduction) can be examined. This approach was used during our experiments to aid in the selection of e_{max}.
3) Consider the case of the standard normal Gaussian window, $e_{max} > 2$, and $H = h\hat{\Sigma}^{0.5}$ where h is the width parameter and $\hat{\Sigma}^{0.5}$ is the symmetric square-root of the empirical covariance matrix. In this case, since e_{max} is large, the training algorithm will terminate with a single reference vector which is the sample mean, $\hat{\mu}$, of the training set. Now $m = 1$, $w_1 = n$, $\mathbf{r}_1 = \hat{\mu}$, which when substituted into (2.2) results in a Gaussian density function with an h-dependent covariance matrix. By using a suitable method to select $h = 1$, (2.2) becomes $p(\mathbf{x}) \sim N[\hat{\mu}, \hat{\Sigma}]$. Therefore, the WPW procedure can be viewed as a continuous bridge between the fully nonparametric PW classifier ($e_{max} = 0$) and the fully parametric Gaussian classifier ($e_{max} > 2$).

3 Experimental Results

Experiments were conducted with synthetic and real data to show the usefulness of the weighted-Parzen-window (WPW) approach. In all cases, standard normal Gaussian window functions were used. The value of H was either equal to $h\hat{\Sigma}^{0.5}$ or hI, where I is the d-by-d identity matrix. Also, the a priori probabilities were set equal for each class. Selection of h was accomplished by varying it over several orders of magnitude, finding the *leave-one-out* (LV) error rate of the Parzen-window (PW) classifier, and choosing the value h_{opt} corresponding to the minimum [6]. This value was used for designing both the PW and WPW classifiers. Selection of e_{max} is discussed below. In the following sections N and M are used to denote the total number of training samples and reference vectors for all classes, respectively.

A two class bivariate data set was synthesized to demonstrate the WPW approach. The data were first drawn from a standard normal distribution, and then various mean vectors were added to the data so that the first class is bimodal while the second is unimodal, centered between the modes of the first. The *Bayes error* rate was estimated by using the optimal classifier to label 10,000 samples from each class. The error rate was determined to be 5.4%. To facilitate graphing, the data set's size was kept moderate with 200 total samples. Fig. 2a shows the data with the Bayes-optimal decision boundaries. Fig. 2b shows the LV error rate for the PW classifier as a function of the width parameter. From this curve h_{opt} was selected as 1.0. The effect of e_{max} was studied by varying it from 0 to 2 in increments of 0.01. For each value of e_{max}, the LV error rate for the WPW classifier and the average number of total reference vectors, $\overline{M}$, was recorded. (Note that $N = 200$ classifiers were designed with $N - 1$ training samples for each value of e_{max}, so the maximum value of $\overline{M}$ is 199.) Fig. 2c shows the LV error rate as a function of the average number of reference vectors and the corresponding error parameters, $0 \leq e_{max} \leq 0.5$. Note that a considerable reduction was achieved without introducing significant classification error. Furthermore, the curves of Fig. 2c describe well the WPW classifier; i.e., for any desired error parameter, e_{max}, we know the average number of reference vectors, $\overline{M}$, and the LV error rate. Fig. 2d shows the WPW weighted reference vectors and the decision boundaries that resulted when $e_{max} = 0.1$. These decision boundaries approximate the Bayes boundaries and are very similar to the PW boundaries (not shown).

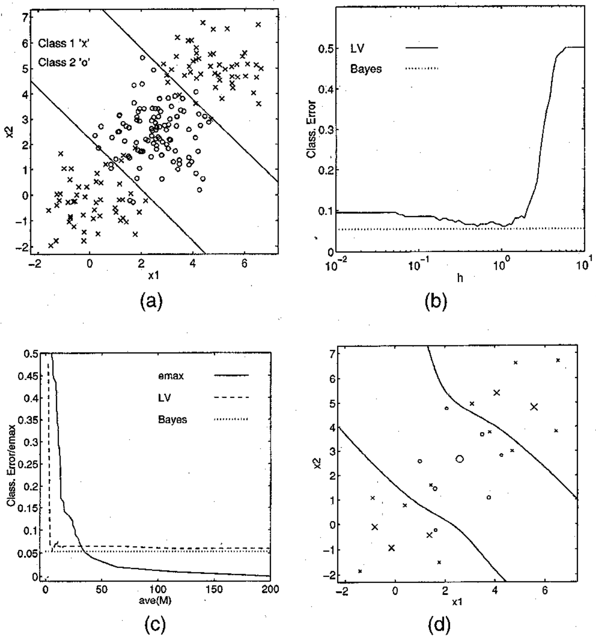

Fig. 2. Results of the WPW algorithm for synthetic data.

To compare the WPW approach with the reduced Parzen classifier, the experiment of reference [5], p. 555, was replicated. This experiment used a synthetic, two-class, eight-dimensional non-normal data set with a known Bayes error rate of 7.5% [5]. The experiment demonstrates how the *holdout* error rate [5] varies as the size of the classifier is reduced. (For the WPW, this procedure required modification of the training algorithm so that it would terminate for a particular value of m.) Fig. 3 shows the results for the WPW classifier. Note that the error curve is the average of the 10 trials and that the error bars indicate one standard deviation. These results are comparable to those of the reduced Parzen classifier [5], p. 556, except that improved performance is noted in the case of the WPW for values of $1 < m \leq 6$. (Fig. 3 has been drawn in the same manner as the figure shown in [5] p. 556, to facilitate visual comparison.)

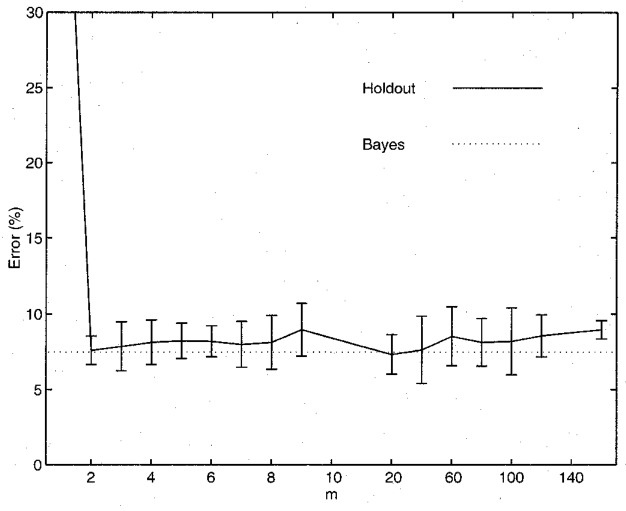

Fig. 3. Classification error as a function of m for the WPW classifier and non-normal data.

Experiments were conducted with three real data sets. The first two data sets, Iris and IMOX, have been widely referenced in the literature. The Iris data are four kinds of measurements for three classes of iris with 50 samples in each class [3]. The IMOX data consists of eight measurements for 48 handwritten characters in each class which are I, M, O, and X [6]. The third data set used was extracted from the acoustic signature of a laser-welder. A single feature is used for two classes of weld which correspond to full- and partial- penetration. This data set is large, with 1,007 training samples for the two classes. The following procedure was used to select e_{max} for each of the real data sets. First, e_{max} was selected large, greater than two, so that the WPW training algorithm would terminate with one reference vector. During training, e and the cluster indices were recorded for every training step. Then, e was plotted as a function of the number of reference vectors for each class (Fig. 4). The curve was then used to select a suitable value of e_{max}. For example, in the case of the IMOX and Iris data, we see that the curves are relatively level for $e < 0.1$, which indicates that a suitable value for e_{max} would be approximately 0.1. Similarly, in the case of the laser data, a suitable value of e_{max} would also be approximately 0.1. However, in this case, the reduction of the training set is much more dramatic. Once e_{max} is selected, the WPW classifier can easily be implemented by using the cluster indices that were previously stored. Using Fig. 4, several values of e_{max} were selected. Table 1 shows the results for the given values of e_{max} and h_{opt}. For each data set, the PW and WPW leave-one-out error rates are shown. Also, the storage reduction is shown in the case of the WPW. Again, note that $M \leq N - 1$ for N total training samples. In all cases shown in Table 1, significant reduction is observed, and in most cases, there is little or no change in error rate. Note that the WPW laser-weld classifier was designed with less than 2.0% of the original training samples while maintaining the same error rate as the PW classifier.

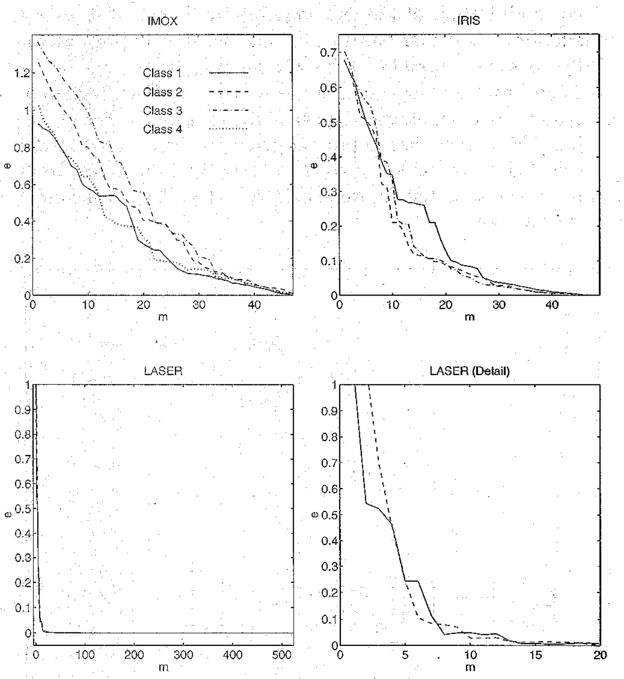

Fig. 4. Error (e) as a function of the training level for three sets of real data.

TABLE 1
EXPERIMENTAL RESULTS FOR THE REAL DATA

Data	h_{opt}	PW Error	e_{max}	WPW Error	$\text{ave}\left(\frac{M}{N-1}\right)$
Iris	1.30	0.0133	0.10	0.0133	0.402
			0.20	0.0200	0.290
			0.30	0.0200	0.214
IMOX	1.05	0.0521	0.10	0.0521	0.724
			0.20	0.0521	0.562
			0.30	0.0573	0.494
Laser	0.70	0.0745	0.05	0.0745	0.019
			0.10	0.0745	0.015
			0.15	0.0735	0.013

Error rates found by leave-one-out technique.

4 CONCLUSION

This correspondence introduced the *weighted-Parzen-window* (WPW) classification approach. This technique uses a *clustering* procedure to find a smaller set of reference vectors and weights which are used to approximate the *Parzen-window* (PW) classifier. The WPW approach is used to reduce the computational burden and storage requirements for the classifier. The WPW procedure uses the PW estimate as a baseline during the training phase which is terminated when the distance between the two estimates exceed a threshold, e_{max}. Procedures for selecting e_{max} were proposed. Experimental results showed that significant reductions could be achieved with only minimal, if any, error rate degradation for synthetic and real data sets.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support provided by the Pennsylvania State University's Applied Research Laboratory, who, under a U.S. Office of Naval Research grant, partially funded this work. Part of this work was also funded by National Science Foundation grant IRI-9309100. We thank Professors N.K. Bose, L.H. Sibul, and R.L. Tutwiler for their valuable comments. Furthermore, we gratefully acknowledge D.F. Farson and K.L. Hillsley, who provided the laser-weld data, and A.K. Jain who supplied the IMOX data. Also, we thank the anonymous reviewers whose thoughtful comments helped to greatly enhance this correspondence. Finally, we thank A.G. Pisani for proofreading the manuscript.

REFERENCES

[1] M.R. Anderberg, *Cluster Analysis for Applications.* New York: Academic Press, 1973.
[2] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis.* New York: John Wiley & Sons, 1973.
[3] R.A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics,* vol. 7, pp. 179-188, 1936.
[4] J. Fan and J.S. Marron, "Fast implementation of nonparametric curve estimators," *J. Computational and Graphical Statistics,* vol. 3, no. 1, pp. 35-56, 1994.
[5] K. Fukunaga, *Statistical Pattern Recognition,* 2nd edition. San Diego, Calif: Academic Press Inc., 1990.
[6] A.K. Jain and M.D. Ramaswami, "Classifier design with Parzen windows," *Pattern Recognition and Artificial Intelligence,* pp. 211-228, E.S. Gelsema and L.N. Kanal, eds. North-Holland: Elsevier Science Publishers B.V., 1988.
[7] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statistics,* vol. 33, pp. 1,065-1,076, Sept. 1962.
[8] M. West, "Approximating posterior distributions by mixtures," *J. Royal Statistical Soc. B,* vol. 55, no. 2, pp. 409-422, 1993.