Neural Networks, Vol. 9, No. 5, pp. 787–796, 1996

CONTRIBUTED ARTICLE

Sequential Competitive Learning and the Fuzzy *c*-Means Clustering Algorithms

NIKHIL R. PAL,[1] JAMES C. BEZDEK[2] AND RICHARD J. HATHAWAY[3]

[1]Indian Statistical Institute, [2]The University of West Florida and [3]Georgia Southern University

(*Received* 7 *October* 1994; *revised and accepted* 5 *July* 1995)

Abstract—*Several recent papers have described sequential competitive learning algorithms that are curious hybrids of algorithms used to optimize the fuzzy c-means (FCM) and learning vector quantization (LVQ) models. First, we show that these hybrids do not optimize the FCM functional. Then we show that the gradient descent conditions they use are not necessary conditions for optimization of a sequential version of the FCM functional. We give a numerical example that demonstrates some weaknesses of the sequential scheme proposed by Chung and Lee. And finally, we explain why these algorithms may work at times, by exhibiting the stochastic approximation problem that they unknowingly attempt to solve.*

Keywords—Alternating optimization, Fuzzy *c*-means, Gradient-based fuzzy *c*-means, Grouped coordinate minimization.

1. INTRODUCTION

Clustering in unlabeled object data set $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathcal{R}^p$ is the assignment of (hard *or* fuzzy *or* probabilistic) label vectors to the vectors in X, and hence, to the objects generating X. *c-partitions* of X are sets of (cn) values $\{u_{ik}\}$ that are arrayed as a $(c \times n)$ matrix $U = [u_{ik}]$:

$$M_{fcnu} = \left\{ U \in \mathcal{R}^{cn} \middle| 0 \leqslant u_{ik} \leqslant 1 \; \forall \, i, \; k; \; \forall k, \;\; u_{ik} > 0 \; \exists \, i; \right.$$

$$\left. 0 < \sum_{k=1}^{n} u_{ik} < n \; \forall \, i \right\}; \tag{1a}$$

$$M_{fcn} = \left\{ U \in M_{fcnu} \middle| \sum_{i=1}^{c} u_{ik} = 1 \; \forall \, k \right\}; \tag{1b}$$

$$M_{cn} = \{ U \in M_{fcn} | u_{ik} = 0 \text{ or } 1 \; \forall \, i \text{ and } k \}. \tag{1c}$$

Equations (1) define, respectively, the sets of unconstrained, constrained, and crisp *c-partitions* of X. Each column of U in $M_{fcnu}(M_{fcn}, M_{cn})$ is a label vector. The reason these matrices are called *partitions* follows from the interpretation of their entries. If U is fuzzy, u_{ik} is taken as the *membership* of $\mathbf{x}_k$ in the ith partitioning fuzzy subset (cluster) of X. If U is probabilistic, u_{ik} is usually the (posterior) probability $p(i|\mathbf{x}_k)$ that, given $\mathbf{x}_k$, it came from class i.

This paper discusses algorithms that are related to approximate minimization of the *Fuzzy c-means* (FCM) functional, which defines a constrained, non-linear optimization problem with both equality and inequality constraints (Bezdek, 1981). For a given set of unlabeled object data X, the problem is

$$\underbrace{\min}_{(U, \mathbf{V})} \left\{ J_m(U, \mathbf{V} : X) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m \|\mathbf{x}_k - \mathbf{v}_i\|_A^2 \right\}, \tag{2}$$

where $U \in M_{fcn}$ is a constrained fuzzy c-partition of X, $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c) \in \mathcal{R}^{cp}$ is a vector of (unknown) cluster centers (weights or prototypes), $\mathbf{v}_i \in \mathcal{R}^p$ for $1 \leqslant i \leqslant c$, $m \in [1, \infty)$ modifies the weight of each fuzzy membership and $\|\mathbf{x}\|_A = \sqrt{\mathbf{x}^T A \mathbf{x}}$ is any inner product norm. The first method discussed for the solution of (2) was approximate minimization of J_m by the FCM algorithm, which is based on (batch) iteration through first order necessary conditions for its local extrema. Problem (2) for $m = 1$, $U \in M_{cn}$ is the well-known *hard c-means* (HCM) model.

Recently, Park and Dagher (1994) and Chung and

Lee (1992), have proposed sequential competitive learning algorithms that are claimed to be related to problem (2). The first purpose of this note is to dispel this idea. Then we will show that, instead, their algorithms can be derived from an optimization problem that is related (but not equivalent) to (2) in the sense that one (instead of all c) term of (2) is examined. Next, we numerically compare Chung and Lee's algorithm to some outputs obtained by FCM on the IRIS data. And finally, we exhibit the optimization problem that implicitly drives these two algorithms.

The FCM algorithm described later is best understood and analyzed using the framework of *grouped (variable) coordinate minimization* (GCM), which is a general iterative method for solving minimization problems. Suppose we wish to solve the problem

$$\underbrace{\min}_{\Omega}\{f(\mathbf{x})\}$$

where $f: \Omega \subset \mathcal{R}^{s+q} \rightarrow \mathcal{R}$. In many cases this cannot be done directly, and the problem is reformulated in terms of groups of subvariables. Following the approach in Bezdek et al. (1987) and Hathaway et al. (1991), we wish to solve

$$\textit{Problem } \mathcal{O} : \underbrace{\min}_{\Omega_1 \times \Omega_2}\{f(\mathbf{y}, \mathbf{z})\}, \tag{3}$$

over some feasible region $\Omega = \Omega_1 \times \Omega_2$, where $f: \Omega = \Omega_1 \times \Omega_2 \subset \mathcal{R}^{s+q} \rightarrow \mathcal{R}$ is a function of the vector variables $\mathbf{y} \in \Omega_1 \in \mathcal{R}^s$ and $\mathbf{z} \in \Omega_2 \in \mathcal{R}^q$. Partitioning of the set of independent variables into $\mathbf{y}$ and $\mathbf{z}$ parts can be done in many ways. In its most naive form, each component of both sets of variables is grouped as a singleton. In this case problem (3) becomes

$$\underbrace{\min}_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_{s+q}} \{f(x_1, x_2, \ldots, x_{s+q})\},$$

and in this form is called the method of coordinate descent (Luenberger, 1965). A useful partitioning for GCM is one in which the resulting minimization of each vector of coordinates in (3) can be done in a computationally efficient manner.

The exact-minimization version of GCM to solve (3) generates the $t+1$th iterate $(\mathbf{y}_{t+1}, \mathbf{z}_{t+1}) \in \mathcal{R}^s \times \mathcal{R}^q$ from the current iterate $(\mathbf{y}_t, \mathbf{z}_t)$ according to the GCM update equations:

Exact GCM update: $(\mathbf{y}_t, \mathbf{z}_t) \rightarrow (\mathbf{y}_{t+1}, \mathbf{z}_{t+1})$ as follows

$$\textit{Problem } \mathcal{A} : \quad \mathbf{z}_{t+1} = \underbrace{\arg\min}_{\mathbf{z} \in \Omega_2} \{f(\mathbf{y}_t, \mathbf{z})\}; \text{ and} \tag{4a}$$

$$\textit{Problem } \mathcal{B} : \quad \mathbf{y}_{t+1} = \underbrace{\arg\min}_{\mathbf{y} \in \Omega_1}\{f(\mathbf{y}, \mathbf{z}_{t+1})\}. \tag{4b}$$

Problem $\mathcal{A}$ and $\mathcal{B}$ are sub-problems of problem $\mathcal{O}$, and were it possible to solve problem $\mathcal{O}$ directly in terms of the joint variables $(\mathbf{y}, \mathbf{z})$, we would not be interested in them. Each of these problems may itself be unconstrained or constrained, depending on the nature of the feasible sub-regions Ω_1 and Ω_2. Windham (1987) referred to solving each of the "half" problems $\mathcal{A}$ and $\mathcal{B}$ as *alternating optimization* (AO) between the *half-steps* defined at (4). It often happens that a direct solution to problem $\mathcal{O}$ is unavailable. This is the case with the FCM model (2), as well as, for another example, maximization of the likelihood of X when the data are assumed to be drawn from a mixture of normal distributions. The algorithm for this latter example is the well known *expectation-maximization* (EM) algorithm (Redner et al., 1987). In such cases, GCM as written at (4) provides one of several alternatives for attempting solutions to (3). There are three cases for eqns (4):

Case 1. *Explicit (analytically exact) Half-steps.* If (4a) and (4b) are explicitly defined in terms of each other, their half-steps are analytically exact. We refer to explicit half-steps in (4) as *analytically exact half-steps* of GCM iteration. Formally, this case corresponds to the situation where we have explicit formulae for the new joint iterate $(\mathbf{y}_{t+1}, \mathbf{z}_{t+1})$ in terms of each half-iterate, i.e.,

$$\mathbf{z}_{t+1} = \Phi(\mathbf{y}_t); \text{ and} \tag{5a}$$

$$\mathbf{y}_{t+1} = \Psi(\mathbf{z}_{t+1}). \tag{5b}$$

Φ and Ψ in (5) are known in closed form. The usual method of finding equations like (5) is via calculus based optimization; unconstrained, LaGrange multiplier or Kuhn–Tucker theory, depending on the nature of Ω_1 and Ω_2. Although not needed for the discussion below, we mention the other two cases for the sake of completeness, and refer readers to Hathaway and Bezdek (1991) for more details.

Cases 2 *and* 3. *Implicit (numerically exact and numerically inexact) Half-steps.* If either Φ or Ψ (or both) are implicitly defined in terms of each other, one (or both) of the sets of variables in (4) must be found numerically. Generally, the form of implicit definition may suggest a numerical technique, for example, linear programming, Newton's method, gradient descent, simulated annealing, etc. In any case, we differentiate between two sub-cases: there are "numerically exact" and "numerically inexact"

numerical solutions. An implicit numerical half-step is *numerically exact* where the solution is computed exactly using a numerical procedure that terminates only at the solution; otherwise, we call the half-step *numerically inexact*. The distinction between these two cases can be significant. For example, a numerically exact half-step might require 100 iterations of Newton's method; whereas one iteration constitutes a numerically inexact half-step. It has been shown by Bezdek et al. (1987) that under fairly general conditions, numerical inexact and numerically exact half-steps are entirely equivalent for producing a q-linear rate of convergence[1] of $\mathbf{y}_t, \mathbf{z}_t$, so the intermediate case (numerically exact solution) is often not needed—the potential savings in computational complexity by using numerically inexact half-steps is obvious.

2. THE FCM ALGORITHM

Optimization of J_m is not amenable to direct solution in the joint variables $(U, \mathbf{V})$. The AO approach to problem (2) was first formulated in form (4) by Dunn (1974) for the special case $m = 2$, and in the same year, Bezdek (1973) described the general AO problem derived from (2) for any $m \geqslant 1$. The first order necessary conditions are contained in the following:

Fuzzy c-means (FCM) Theorem (Bezdek, 1981): Assume $m > 1$ and $\|\mathbf{x}_k - \mathbf{v}_i\|_A^2 > 0$, $1 \leqslant i \leqslant c$; $1 \leqslant k \leqslant n$. $(U, \mathbf{V}) \in M_{fcn} \times \mathscr{R}^{cp}$ may minimize J_m only if:

Solution $\mathscr{A} - \mathscr{FCM}$:

$$U_{t+1} = \Phi(V_t; X) = [u_{ik,\,t+1}] = \left[\sum_{j=1}^{c} \left(\frac{\|x_k - v_{i,\,t}\|_A}{\|x_k - v_{j,\,t}\|_A}\right)^{\frac{2}{m-1}}\right]^{-1} \quad (\forall\, i,k); \qquad (6a)$$

[1] $\{x_k\}$ converges q-linearly to x^* if

$$\underbrace{\lim}_{k\to\infty} |x_k - x^*| = 0,$$

and there is a constant c in [0, 1) and an integer $k_0 > 0$ such that, for all $k \geq k_0$, $|x_{k+1} - x^*| \leq c|x_k - x^*|$.

Solution $\mathscr{B} - \mathscr{FCM}$:

$$\mathbf{V}_{t+1} = \Psi(U_{t+1}; X) = \left\{ \mathbf{v}_{t+1} = \frac{\sum_{k=1}^{n} (u_{ik,t+1})^m \mathbf{x}_k}{\sum_{k=1}^{n} (u_{ik,t+1})^m}; 1 \leqslant i \leqslant c \right\}. \qquad (6b)$$

We have put eqns (6) in the format of (5), because the necessary conditions for U and $\mathbf{V}$ shown at (6) are analytically exact half-iterates. Problem $\mathscr{A}$-$\mathscr{FCM}$ is constrained by (1b), and solution $\mathscr{A}$-$\mathscr{FCM}$ is derived by zeroing the gradient of the Lagrangian of Φ with respect to *one column* of U (this works because J_m is a sum of n non-negative terms, so the minimum of the sum is the sum of the minima). This does not automatically enforce the inequality constraints on the u_{ik}, but, fortuitously, they are satisfied by (6a) anyway. Differentiability of Φ in the u_{ik} is guaranteed by using an inner product norm in (2).

Problem $\mathscr{B}$-$\mathscr{FCM}$ is unconstrained, and solution $\mathscr{B}$-$\mathscr{FCM}$ (6b) is easily obtained by setting the gradient of Ψ with respect to $\mathbf{v}_i$ equal to the zero vector in $\mathscr{R}^p$. Singularity in FCM occurs when one or more of the distances $\|\mathbf{x}_k - \mathbf{v}_i\|_A^2 = 0$ at any iterate. In this case (rare in practice), (6a) cannot be calculated. When this happens, assign 0's to each non-singular class, and distribute memberships to the singular classes arbitrarily subject to constraint (1b). Although this theorem is stated for $m > 1$, it is actually true for $m \geqslant 1$, because conditions (6) converge to the well known necessary conditions for (batch) HCM, and $J_m \to J_1$ as $m \to 1$ from above (Bezdek, 1981). Table 1 gives a brief specification of the FCM algorithms as used in our example below.

Iteration simply loops through one cycle of estimates for $\mathbf{V}_{t-1} \to U_t \to \mathbf{V}_t$ and then checks $\|\mathbf{V}_t - \mathbf{V}_{t-1}\| \leqslant \varepsilon$. Equivalently, the entire procedure can be shifted one half cycle, so that initialization is done on U_0, and the iterates become $U_{t-1} \to \mathbf{V}_t \to U_t$, with the alternate termination criterion $\|U_t - U_{t-1}\| \leqslant \varepsilon$. The literature contains both specifications; the convergence theory is the same in either case. There are some obvious advantages to the form given here in terms of speed

TABLE 1
Fuzzy c-Means (FCM) Algorithms (Bezdek, 1981)

Store	Unlabeled object data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathscr{R}^p$
Pick	☛ $1 < c < n$ ☛ $1 < m < \infty$ ☛ T = *max. iterations*, $T \in 1, 2, \ldots$ ☛ $0 < \varepsilon$ ☛ Norm for J_m: $\langle \mathbf{x}, \mathbf{x} \rangle_A = \|\mathbf{x}\|_A^2 = \mathbf{x}^T A \mathbf{x}$ ☛ Norm for $E_t = \|\mathbf{V}_t - \mathbf{V}_{t-1}\|$
Guess	$\mathbf{V}_0 = (\mathbf{v}_{1,0}, \mathbf{v}_{2,0}, \ldots, \mathbf{v}_{c,0}) \in \mathscr{R}^{cp}$
Iterate	For $t = 1$ to T: Calculate $\mathbf{U}_t$ with $\mathbf{V}_{t-1}$ and (6a) Calculate $\mathbf{V}_t$ with U_t and (6b) If $E_t = \|\mathbf{V}_t - \mathbf{V}_{t-1}\| \leqslant \varepsilon$, stop and put $(U_t, \mathbf{V}_t) = (U_t, \mathbf{V}_t)$; else Next t
Use	Prototypes $\mathbf{V}_t$ and/or fuzzy labels U_t

and storage. The alternative form that terminates on U's is more stringent, since more parameters must become close before termination is achieved, and it can happen that different results ensue by using the same ε with both forms. FCM has five algorithmic parameters (U_0 *or* $\mathbf{V}_0, A, c, m, \varepsilon$). Variation in any of these parameters can affect the output of FCM for a fixed set of unlabeled data. Consideration of these effects on the outputs of FCM, as well as its convergence and convergence rates, are discussed in Bezdek (1981).

Another aspect of FCM is speed. The AO approach to minimizing J_m involves $c(n+p)$ variables, and for large data sets (a typical image, for example, has $n = 256 \times 256 = 65{,}536$), FCM can be slow. Another way to approach problem (2) is to formulate it using the necessary conditions at (6) to eliminate one of the two sets of grouped coordinates. Since U will almost always contain more variables than $\mathbf{V}$, we will usually want to reformulate (2) by substitution of (6a) into the formula for J_m:

$$
\begin{aligned}
J_m(U,\mathbf{V}:X) &= \sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m\|\mathbf{x}_k-\mathbf{v}_i\|_A^2=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m d_{ikA}\\
&=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})(u_{ik})^{m-1}d_{ikA}\\
&=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})\left[\frac{1}{\sum_{j=1}^{c}\left(\frac{d_{ikA}}{d_{jkA}}\right)^{\frac{1}{m-1}}}\right]^{m-1}\\
&\times d_{ikA}=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})\left[\frac{\frac{1}{d_{ikA}}d_{ikA}}{\sum_{j=1}^{c}\left(\frac{1}{d_{jkA}}^{\frac{1}{m-1}}\right)^{m-1}}\right]\\
&=\sum_{k=1}^{n}\left[\frac{1}{\left(\sum_{j=1}^{c}\left(\frac{1}{d_{jkA}}\right)^{\frac{1}{m-1}}\right)^{m-1}}\right]\\
&\times\underbrace{\left(\sum_{i=1}^{c}u_{ik}\right)}_{=1}=\sum_{k=1}^{n}\left[\frac{1}{\left(\sum_{j=1}^{c}\left(\frac{1}{d_{jkA}}\right)^{\frac{1}{m-1}}\right)^{m-1}}\right]\\
&=R_m(\mathbf{V}:X). \qquad (7)
\end{aligned}
$$

In (7), $\|\mathbf{x}_k-\mathbf{v}_i\|_A^2=d_{ikA}$. This calculation was first done by Bezdek (1976) for a somewhat different purpose, but as we shall see, it is relevant for the present article. Minimization of $R_m(\mathbf{V}:X)$ is unconstrained in the variables $\mathbf{V}$, and it is often easier to solve this equivalent problem with commercially available software (Hathaway & Bezdek, 1995). It may seem that elimination of the U variables has also eliminated our chance of describing the cluster memberships of the data, but the partitioning U can be easily computed using the minimizing $\mathbf{V}^*$ in (6a).

It is important to understand why we say the minimization of $R_m(\mathbf{V}:X)$ is equivalent to minimization of $J_m(U,\mathbf{V}:X)$.[2] This is because of the relationship between minimizers of the original and reformulated criteria. If all of the $d_{ikA}>0$, it is easy to show (see the Appendix) that for some U^*, $(U^*,\mathbf{V}^*)$ is a global (local) minimizer of $J_m(U,\mathbf{V}:X)$ if and only if $\mathbf{V}^*$ is a global (local) minimizer of $R_m(\mathbf{V}:X)$. We can drop the assumption $d_{ikA}>0$ and still have these implications if we extend the definition of R to allow $d_{ikA}=0$. In practice, this can be done by extending the arithmetic in R's calculation to include $1/0=\infty$ and $1/\infty=0$. This extension is automatically done in a computing environment such as MATLAB.

3. SEQUENTIAL FUZZY COMPETITIVE LEARNING

There have been several recent attempts to fuzzify sequential competitive learning (CL) algorithms such as Kohonen's learning vector quantization (LVQ) (Kohonen, 1989). Chung and Lee (1992) discussed an algorithm they call unsupervised fuzzy competitive learning (UFCL). Park and Dagher (1994) independently presented an algorithm called gradient based fuzzy c-means (GBFCM). Both of these algorithms start by dropping all but the kth term in (2), resulting in the reduced optimization problem:

$$
\underbrace{\min}_{(\mathbf{U}_k,V)}\left\{J_{m,k}(\mathbf{U}_k,\mathbf{V}:\mathbf{x}_k)=\sum_{i=1}^{c}(u_{ik})^m\|\mathbf{x}_k-v_i\|_A^2\right\}, \qquad (8)
$$

where $\mathbf{U}_k$ is a constrained fuzzy label vector in $N_{fc}=\{\mathbf{y}\in\mathcal{R}^c: 0\leqslant y_i\leqslant 1\forall i;\sum_{i=1}^{c}y_i=1\}$—that is, one column of U in M_{fcn}, $\mathbf{V}=(\mathbf{v}_1,\mathbf{v}_2,\ldots,\mathbf{v}_c)\in\mathcal{R}^{cp}$ is a vector of (unknown) cluster centers (weights or prototypes), $\mathbf{v}_i\in\mathcal{R}^p$ for $1\leqslant i\leqslant c$, $m\in[1,\infty)$ modifies the weight of each fuzzy membership and $\|\mathbf{x}\|_A=\sqrt{\mathbf{x}^TA\mathbf{x}}$ is any inner product norm.

Problem (8) is explicitly stated by eqns (6) and (7) by Park and Dagher (1994) for the special case of $m=2$ and the Euclidean norm (A the $p\times p$ identity matrix). Problem (8) appears in two forms in Chung and Lee (1992). First, as the point of departure for derivation of update equation (2) in Chung and Lee (1992) for the prototypes in the unsupervised (crisp) competitive learning algorithm (Chung and Lee's UCL), where our $J_{1,k}$ is called $J_k(W,V)$ on p. 541.

[2] See our remark following the theorem given in the Appendix.

UCL), where our $J_{1,k}$ is called $J_k(W, V)$ on p. 541. And our (8) appears again in Chung and Lee (1992) preceding the derivation of their update equation (9) for their UFCL scheme, our $J_{m,k}$ being called $J_k^m(V)$. The Euclidean norm and $m \geqslant 1$ are used in Chung and Lee.

We discuss (8) in the form shown, since everything we have to say about it is true for the more general cases that include m and A as parameters. Since $J_{m,k}(\mathbf{U}_k, \mathbf{V} : \mathbf{x}_k)$ is positive definite, its global minimum is zero, achieved by setting, for any U_k in N_{fc}, $\mathbf{x}_k = \mathbf{v}_i$ for all i such that $u_{ik} \neq 0$. If this fact goes unnoticed, we might again turn to AO as a means for solution of (8). Setting up this new problem as the pair of subproblems $\mathscr{A}$-$\mathscr{SFCM}$ and $\mathscr{B}$-$\mathscr{SFCM}$, which for convenience we call *sequential fuzzy c-means* (SFCM), exactly as before will lead to the following:

"Theorem" SFCM: Assume $m > 1$ and $\|\mathbf{x}_k - \mathbf{v}_{i,t}\|_A^2 > 0$, $1 \leqslant i \leqslant c$; $1 \leqslant k \leqslant n$. The AO minimizer $(\mathbf{U}_{k,t+1}, \mathbf{V}_{t+1}) \in N_{fc} \times \mathscr{R}^{cp}$ of $J_{m,k}$ is:

Solution $\mathscr{A} - \mathscr{SFCM}$:

$$\mathbf{U}_{k,t+1} = \Phi(\mathbf{V}_t; \mathbf{x}_k) = [u_{ik,t+1}]$$

$$= \left[\sum_{j=1}^{c} \left(\frac{\|\mathbf{x}_k - \mathbf{v}_{i,t}\|_A}{\|\mathbf{x}_k - \mathbf{v}_{j,t}\|_A}\right)^{\frac{2}{m-1}}\right]^{-1} \quad \forall i; \tag{9a}$$

Solution $\mathscr{B} - \mathscr{SFCM}$:

$$\mathbf{V}_{t+1} = \Psi(\mathbf{U}_{k,t+1}; \mathbf{x}_k) = \{\mathbf{v}_{i,t+1} = \mathbf{x}_k : 1 \leqslant i \leqslant c\}. \tag{9b}$$

Equation (9a), the analytically exact half-iterate derived with Lagrange multipliers that solves problem $\mathscr{A}$-$\mathscr{SCFM}$, is precisely the same as (6a) for the kth column of U because that is how (6a) was originally derived. So, the solutions to problems $\mathscr{A}$-$\mathscr{FCM}$ and $\mathscr{A}$-$\mathscr{SFCM}$ *seem at first* to be the same. However, setting the gradient of Ψ with respect to v_i equal to the zero vector in $\mathscr{R}^p$ to find the necessary condition for problem $\mathscr{B}$-$\mathscr{SFCM}$ yields:

$$\nabla_{\mathbf{v}_i}\left(\sum_{j=1}^{c} u_{jk}^m \|\mathbf{x}_k - \mathbf{v}_j\|_A^2\right) = -2u_{ik}^m A(\mathbf{x}_k - \mathbf{v}_i) = \mathbf{0} \Leftrightarrow \mathbf{x}_k = \mathbf{v}_i. \tag{10}$$

Thus, (9b) has only the trivial solution previously mentioned. Returning to (9a) equipped with (9b) immediately renders conditions (9a) in Theorem SFCM indeterminate, and shows that (9a) is not really necessary—any $\mathbf{U}_{k,t+1}$ in N_{fc} will do. Since Problem (8) has no non-trivial solution, there cannot be a "sequential" version of FCM based on minimizing $J_{m,k}$.

Faced with this dilemma, Park and Dagher (1994) and Chung and Lee (1992) both resort to an alternative tactic for problem $\mathscr{B}$-$\mathscr{SFCM}$. Instead of setting the gradient of Ψ with respect to $\mathbf{v}_i$ equal to the zero vector in $\mathscr{R}^p$, these authors compute the gradient of Ψ, and use it to write an update equation or learning law for the new prototypes based on the current value of $\mathbf{U}_k$ and the method of steepest descent. Here is the calculation:

$$\nabla_{\mathbf{v}_i}\left(\sum_{j=1}^{c} u_{jk}^m \|\mathbf{x}_k - \mathbf{v}_j\|_A^2\right) = -2u_{ik}^m A(\mathbf{x}_k - \mathbf{v}_i) \tag{11a}$$

$$= -\eta u_{ik}^m(\mathbf{x}_k - \mathbf{v}_i) \text{ for } A = I, \text{ the Euclidean norm case.} \tag{11b}$$

The constant 2 is simply absorbed into η in (11b). Using the general form of the method of gradient descent now gives with (11b) the update rule:

$$\mathbf{v}_{i,t+1} = \mathbf{v}_{i,t} + \eta u_{ik}^m(\mathbf{x}_k - \mathbf{v}_{i,t})$$

where η is a small positive constant. (12)

Equation (11b) is (9) in Chung and Lee (1992); and it becomes (9) in Park and Dagher (1994) upon taking $m = 2$ (there is a sign change because Park and Dagher reverse the data and prototypes under the norm operation). Both sets of authors couple eqn (12) with our (9a) and this pair of equations defines UCL ($m = 1$) or UFCL ($m > 1$) in Chung and Lee (1992); or GBFCM in Park and Dagher (1994) with $m = 2$. Quoting from p. 1628 of Park and Dagher (1994): "FCM and GBFCM both have an objective function which tries to minimize the distance between...". Thus, Park and Dagher apparently believe that GBFCM does attempt to minimize $J_{2,k}$, that is, solve problem (8) for $m = 2$. We have shown this to be impossible. Their algorithm, however, is partially related to the well defined optimization problem given in Section 5.

Park and Dagher's update rule is, of course, just a special case of Chung and Lee's for $m = 2$. However, the implementation of GBFCM is somewhat different from UFCL. Indeed, we were unable to replicate GBFCM as given in Park and Dagher (1994) on pp. 1628–1629, as there are many inconsistencies and mistakes (for example, the line $e \leftarrow e + v_i(k+1) - v_i(k)$ apparently adds a difference vector to a real number). Table 2 gives the UFCL algorithm as implemented for the examples below.

Chung and Lee apply the same rationale as Park and Dagher for derivation of their UCL and UFCL schemes, and, for the same reason, neither of these algorithms is related to solutions of our (8), much less to solutions of our (2). However, as is the case for GBFCM, the UFCL method is partially related to the reformulated optimization problem defined by

TABLE 2
Unsupervised Fuzzy Competitive Learning (UFCL) Algorithms (Chung & Lee, 1992)

Store	Unlabeled object data $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathscr{R}^p$
Pick	$\blacksquare\ 1 < c < n\ \blacksquare\ 1 < m < \infty\ \blacksquare\ 0 < \alpha_0 < 1\ \blacksquare\ T =$ *max. iterations* $= 30n$
Guess	$\mathbf{V}_0 = (\mathbf{v}_{1,0}, \mathbf{v}_{2,0}, \ldots, \mathbf{v}_{c,0}) \in \mathscr{R}^{cp}$
	For $t = 1$ to T:
	Choose randomly $\mathbf{x}_k \in X$
	For $i = 1$ to c:
	Compute $u_{ik,t} = \left(\sum_{j=1}^{c} \left(\frac{D_{ikl,t}}{D_{jkl,t}} \right)^{\frac{2}{m-1}} \right)^{-1}$
	Update $\mathbf{v}_{i,t} = \mathbf{v}_{i,t-1} + \alpha_t u_{ik,t}^m [\mathbf{x}_k - \mathbf{v}_{i,t-1}]$
	Next i
	Compute $\alpha_t = \alpha_0(1 - t/T)$
	Next t
	Put $(U_t, \mathbf{V}_t) = (U_T, \mathbf{V}_T)$
Use	Prototypes $\mathbf{V}_t$ and/or fuzzy labels U_t

$R_{m,k}$. We show in Section 5 that both algorithms actually perform a stochastic approximation based on minimizing the expected value of our reformulation functional $R_{m,k}$ extended to all of $\mathscr{R}^p$.

Further, Chung and Lee (1992) state, p. 540, that "It has been pointed out (Kong & Kosko, 1991; Hertz et al., 1991) that the basic form of competitive learning (CL) is nothing more than an on-line version of the hard c-means algorithm". We believe that Chung and Lee misinterpret both of these references. From Kong and Kosko (1991), p. 119: "Pattern recognition theorists first studied the UCL algorithm but called it *adaptive K-means clustering* [10]". Kong and Kosko are correctly referring to the *sequential* hard K-means procedure which was analyzed by MacQueen (1967),[3] and summarized by Pal et al. (1993). Equation (7) in Pal et al. (1993) shows this explicitly, and is just (12) above, with a different choice for learning coefficient η. As for Hertz et al. (1991), we assume that Chung and Lee refer to the objective function at eqn (9.8), p. 222 of Hertz et al. This is clearly a different objective function from Chung and Lee's $J_k^m(V)$, i.e., our $J_{m,k}$. The point is that *none* of these sequential CL schemes can be regarded as an "on-line version" of hard c-means, which is a batch algorithm for approximately minimizing J_1 in (2). Hard c-means and sequential hard c-means are similar in appearance, but can produce very, very different results on the same data (this does NOT say that hard *c-means produces better* results—just different ones; the quality of any clustering is an entirely different topic). We will show that the same is true for batch FCM by numerically comparing it to UFCL.

[3] Duda and Hart (1973), p. 250 put it this way; "This procedure [basic minimum-squared-error] is also closely related to the adaptive sequential procedure of Sebestyn (1962), and to the so-called k-means procedure, whose convergence properties were studied by MacQueen (1967)". Batch hard k (or c) means is the algorithm described in Tou and Gonzalez, (1974), p. 94, or by Bezdek (1981), p. 55. Confusion also arises both over the use of c instead of k; because many writers refer to the sequential version simply as k-means, dropping the word adaptive or sequential.

4. NUMERICAL COMPARISON OF FCM AND UFCL

FCM and UFCL both depend on the choice of fuzziness parameter m, but in very different ways. First, remember that $\sum_{i=1}^{c} u_{ik} = 1$ for each input x_k, and that for a fixed u_{ik}, increasing m decreases $(u_{ik})^m$. When m is small (close to 1), FCM and UFCL both tend to produce label vectors that are almost crisp. If one u_{ik} is close to 1, the update for $\mathbf{v}_i$ in (11) will be much larger than the updates for the other $\mathbf{v}_j$s. If, additionally, the current prototypes in UFCL have an unfavorable geometry compared to the central tendencies of clusters in the data, some UFCL updates will move rapidly towards a cluster, while others may move very little. This effect is illustrated in Figure 1. For points in both X_1 and X_2 memberships in class 1 will be close to 1. When this happens, prototype $\mathbf{v}_1$ in Figure 1 will quickly migrate towards the grand mean of X_1 and X_2, and $\mathbf{v}_2$ will never change very much.

FCM, on the other hand, defers updates for the $\mathbf{v}_j$

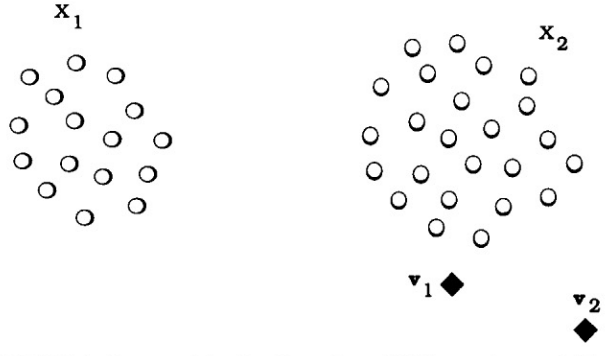

FIGURE 1. A geometric situation where UFCL may be unstable when m is close to 1.

inherently more stable to small values of and changes in m. On the other hand, if m is large (say > 7) all of the u_{ik}'s will be nearly $1/c$ (Bezdek, 1981), and hence all c prototypes will be pulled towards the data point $\mathbf{x}_k$ very slowly by UFCL. This will happen because every $(u_{ik})^m$ will be very small, and every prototype will be updated to almost the same extent. Thus, neither a low nor a high value of m seems desirable in UFCL.

To show that the choice of m can strongly influence the outcome of the UFCL algorithm, we ran UFCL on the IRIS data (Anderson, 1935). IRIS has 150 points in four-dimension (50 points from each of three types of flowers). UFCL was run with m = 1.2, 1.5, 2.0, 2.5, and 7.0. We used $\alpha_0 = 0.3$ (this is one of Chung and Lee's choices) for all five cases. In each case the algorithm was started with the same initial prototype vectors shown in Table 3. Following the suggestion of Chung and Lee the algorithm was run for $T = 30 \times 150 = 4500$ steps. For each case, the same points were used in the same sequence for learning. UFCL did so poorly at $m = 1.2$ and 7.0 with 4500 iterates that we reran these two cases using 20,000 iterates to see if additional learning would drive the final centroids to better positions. From Table 3 it is clear, as illustrated by Figure 1, that for $m = 1.2$ two of the centroids move rapidly, while the third moves very slowly. On the other hand, for $m = 7.0$, all centroids seem to move at a slow but more or less uniform rate. In this latter case 86 points are wrongly classified by the nearest prototype rule, of which 36 are from class 1, although it is well known that class 1 is quite well separated from the other two classes. Neither UFCL nor FCM uses class information during learning. The class information about the physical labels for the IRIS data was used only to produce confusion matrices. The confusion matrices listed in Table 3 are found by applying the nearest prototype classifier to each of the 150 points in IRIS [in all cases we were careful to use the relabeling algorithm described in Pal et al. (1993) to make sure that algorithmic labels were not swapped with their physical counterparts]. The nearest prototype labels are then compared with the actual labels. The ijth entry of the confusion matrix records the number of times real physical label i was instead given label j by the algorithm. For example, three class two points were incorrectly labeled class 1 for $m = 1.5$, and the other 47 class 2 points were incorrectly labeled class 3. Thus, the UFCL-based nearest prototype classifier commits 50, 50, 15, 16, and 86 labeling mistakes, respectively, for m = 1.2, 1.5, 2.0, 2.5, and 7.0. For this example, the best UFCL result in terms of number of missclassifications is obtained for $m = 2$. However, this may not be generally true.

Table 4 contains outputs obtained by FCM on IRIS using exactly the same initializations as shown in Table 3. The termination criterion for FCM was $\varepsilon = 0.000001$. You can see that FCM produces much more stable estimates of the class centroids over the same five choices for m. The results are essentially identical for $m = 1.5$, 2.0, and 2.5; and are very slightly better for $m = 1.2$ and 7.0. Observe that the FCM-based nearest prototype classifier correctly labels all 50 points in class 1 for all five values of m. Compare this with the UFCL results.

The nearest prototype errors derived from the

TABLE 3
Outputs of UFCL on the IRIS Data

Initial Centroids				Final Centroids for $m = 1.2$				Confusion Matrix		
0.9563	0.9570	0.1796	0.8507	5.0146	3.3471	1.5808	0.3007	50	0	0
0.7169	0.4361	0.7920	0.5934	0.4062	0.7713	0.2594	0.5468	3	0	47
0.1419	0.6295	0.1439	0.5399	6.3172	2.9012	4.9575	1.7045	0	0	50
				Final Centroids for $m = 1.5$				Confusion Matrix		
				5.0121	3.3784	1.5203	0.2642	50	0	0
	Init. same as above			1.7779	1.2998	1.1340	0.7011	3	0	47
				6.3125	2.8837	4.9591	1.6919	0	0	50
				Final Centroids for $m = 2.0$				Confusion Matrix		
				4.9984	3.3912	1.4880	0.2469	50	0	0
	Int. same as above			5.9115	2.7784	4.4082	1.4014	0	47	2
				6.7902	3.0636	5.6785	2.0959	0	13	38
				Final Centroids for $m = 2.5$				Confusion Matrix		
				4.9884	3.3816	1.4906	0.2471	50	0	0
	Init. same as above			5.9024	2.7934	4.4110	1.3966	0	45	3
				6.7349	3.0602	5.6148	2.0875	0	13	39
				Final Centroids for $m = 7.0$				Confusion Matrix		
				3.4662	2.0686	2.2071	0.9078	14	36	0
	Init. same as above			5.0748	2.7240	3.1856	1.1355	0	44	6
				4.8528	2.4966	3.2980	1.1330	0	44	6

TABLE 4
Outputs of FCM on the IRIS Data

Initial Centroids				Final Centroids for $m = 1.2$				Confusion Matrix		
0.9563	0.9570	0.1796	0.8507	5.0061	3.4258	1.4657	0.2479	50	0	0
0.7169	0.4361	0.7920	0.5934	5.8921	2.7449	4.3903	1.4272	0	48	2
0.1419	0.6295	0.1439	0.5399	6.8468	3.0742	5.7249	2.0705	0	14	36
				Final Centroids for $m = 1.5$				Confusion Matrix		
				5.0060	3.4203	1.4748	0.2518	50	0	0
Init. same as above				5.8887	2.7485	4.3775	1.4144	0	47	3
				6.8273	3.0662	5.7057	2.0668	0	14	36
				Final Centroids for $m = 2.0$				Confusion Matrix		
				5.0040	3.4141	1.4828	0.2535	50	0	0
Int. same as above				5.8889	2.7611	4.3640	1.3973	0	47	3
				6.7750	3.0524	5.6468	2.0535	0	13	37
				Final Centroids for $m = 2.5$				Confusion Matrix		
				5.0024	3.4077	1.4884	0.2541	50	0	0
Init. same as above				5.8975	2.7770	4.3680	1.3936	0	47	3
				6.7291	3.0440	5.5935	2.0446	0	12	38
				Final Centroids for $m = 7.0$				Confusion Matrix		
				5.0160	3.4015	1.5000	0.2509	50	0	0
Init. same as above				5.9867	2.8452	4.4472	1.4332	0	46	4
				6.5606	3.0039	5.3712	1.9535	0	9	41

The nearest prototype errors derived from the confusion matrices in Tables 3 and 4 are summarized in Table 5, which also shows the number of iterates each algorithm used for each run. We emphasize that UFCL iterates are looks at individual vectors, whereas each FCM iterate is one AO pass through all 150 points. We see that for very low and very high values of m, the UFCL learning process is very slow, and may require a very large number of iterations to get useful results. The error rates speak for themselves. UFCL prototypes are good estimators of central tendency (i.e., good vector quantizers) for the IRIS data in only for two of the five cases, whereas FCM yields stable estimates and very predictable classifier results in all five cases.

Chung and Lee acknowledge in their discussion section that UFCL is critically sensitive to the choice of m, but do not provide any explanation for it. Instead, they suggest a scheme for monotonically decreasing m from, say 2.2 to 1.4 in small steps. This strategy was first suggested for the *fuzzy learning vector quantization* (FLVQ) algorithm described by Bezdek et al. (1992), Bezdek (1992a), and Bezdek (1992b), which is a batch relative of fuzzy c-means. To summarize, we have given a reason for the sensitivity of UFCL (and GBFCM as well) to m, and have also shown that values of m in the range suggested by Chung and Lee can still lead to stability problems.

TABLE 5
Comparison of Error Rates and Number of Iterates

Value of m	UFCL Errors	FCM Errors	UFCL *sequential* Iterates	FCM *batch* Iterates
1.2	50	16	20,000	17
1.5	50	17	4,500	24
2.0	15	16	4,500	26
2.5	16	15	4,500	33
7.0	86	13	20,000	57

5. WHAT DOES UFCL OR GBFCM MINIMIZE?

Suppose we apply the reformulation trick which led to $R_m(\mathbf{V} : X)$ at (7) to the functional $J_{m,k}$, by substituting the necessary condition at (9a) into $J_{m,k}(U, \mathbf{V} : \mathbf{x}_k)$. Doing this simply strips the sum over k from 1 to n from the calculations in (7), leaving us with:

$$J_{m,k}(U, \mathbf{V} : x_k) = \frac{1}{\left(\sum_{j=1}^{c} \left(\frac{1}{d_{jkA}}\right)^{\frac{1}{m-1}}\right)^{m-1}} = R_{m,k}(\mathbf{V} : \mathbf{x}_k). \tag{13}$$

Minimization of $R_m(\mathbf{V} : \mathbf{x}_k)$ is again unconstrained in the variables $\mathbf{V}$. The gradient of $R_{m,k}$ with respect to $\mathbf{v}_i$ is:

$$\nabla_{\mathbf{v}_i}(R_{m,k}(\mathbf{V} : \mathbf{x}_k)) = \nabla_{\mathbf{v}_i}\left(\sum_{j=1}^{c} \left(\frac{1}{d_{jkA}}\right)^{\frac{1}{m-1}}\right)^{-(m-1)} = \ldots$$

$$= -2A\left[\left[\sum_{j=1}^{c} \left(\frac{\|\mathbf{x}_k - \mathbf{v}_{i,t}\|_A}{\|\mathbf{x}_k - \mathbf{v}_{j,t}\|_A}\right)^{\frac{2}{m-1}}\right]^{-m}\right] \times (\mathbf{x}_k - \mathbf{v}_i),$$

where $(\ldots)$ represents some intermediate steps that are not hard to supply. Now, substitution of (9a) for the bracketed expression yields

$$\nabla_{\mathbf{v}_i}(R_{m,k}(\mathbf{V}:\mathbf{x}_k)) = -2Au_{ik}^m(\mathbf{x}_k - \mathbf{v}_i) \qquad (14)$$

Equation (14) is, as it must be, identical to (11a). Zeroing this gradient leads again to only the trivial solution $\mathbf{x}_k = \mathbf{v}_i$ for unconstrained minimization of $R_m(\mathbf{V}:\mathbf{x}_k)$. UFCL and GBFCM do not *iteratively* adjust new estimates of the prototypes using the method of steepest descent based on (14) with $\mathbf{x}_k$ *fixed*, because this would inevitably lead to the trivial global minimum. Instead, they use the steepest descent equation as a one time adjustment to the prototypes, and then move on to the next data point. **At this stage we are in a position to state definitively that UFCL and GBFCM are not solving optimization problems based on either J_m (batch FCM) or $J_{m,k}$ (sequential FCM) in either their original or reformulated versions.**

And yet, *in some cases* UFCL (and hence, GBFCM as well) produces reasonable solutions. Our next task is to see if we can explain why this might be. The derivation of CL algorithms that have learning laws of the general form of the update rule for steepest descent begins by letting $\mathbf{x} \in \mathcal{R}^p$ be a stochastic input vector distributed according to some time invariant probability distribution $f(\mathbf{x})$ for the vector-valued random variable X Tsypkin (1973). Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, \ldots,\}$ be a set of samples drawn from $f(\mathbf{x})$ at time instants $t = 1, 2, \ldots, n, \ldots$. Our objective is to find a set c of prototypes $\mathbf{V} = \mathbf{v}_i$ such that the expected value of $R_m(\mathbf{V}:\mathbf{x})$ is a minimum. In other words, we want to minimize

$$E(R_m(\mathbf{V}:\mathbf{x})) = \underbrace{\int\int\ldots\int}_{\mathbf{x}\in\mathcal{R}^p} R_m(\mathbf{V}:\mathbf{x}) f(\mathbf{x}) d\mathbf{x},$$

where E stands for expectation with respect to $\mathbf{X}$. Since the exact form of $f(\mathbf{x})$ is not known, $E(R_m(\mathbf{V}:\mathbf{x}))$ can be approximately minimized, following Tsypkin (1973), by moving the prototypes in the direction of the negative gradient of the random functional $R_m(\mathbf{V}:\mathbf{x})$. This computation results in (14) again, with $\mathbf{x}_k$ replaced by $\mathbf{x}$.

Applying the steepest descent update to the prototypes *each time a point from the process is submitted* is properly called stochastic approximation of the prototypes that minimize $E(R_m(\mathbf{V}:\mathbf{x}))$ with respect to the unknown density function $f(\mathbf{x})$. Most, if not all of the sequential CL schemes we know of, *implicitly* rely on this fact to justify the use of (12) as an update scheme. It is our belief that UFCL and GBFCM (and, for that matter almost all unsupervised versions of LVQ, including FLVQ) sometimes find useful solutions precisely because the data being processed do have this underlying but unknown structure. This—stochastic approximate minimization of $E(R_m(\mathbf{V}:\mathbf{x}))$—is why UFCL and GBFCM sometimes work.

6. CONCLUSIONS

We have discussed UFCL and GBFCM, two sequential clustering algorithms that were presented as being related to fuzzy c-means, and we have shown that this is not true. The authors of both algorithms confused the use of unconstrained optimization on reformulated versions of sequential FCM with stochastic approximation. We have shown numerically, algebraically, and geometrically why the UFCL scheme is sensitive to the parameter m, as was acknowledged by Chung and Lee. Sequential schemes are sometimes attractive, but, like all algorithms, have some weaknesses. Analysis of UFCL has exposed some of its weaknesses, and we caution authors of algorithms of this type to be careful how they interpret statements such as that in Chung and Lee (1992) p. 550,: "Also, other existing CL algorithms, such as the differential CL (Kong & Kosko, 1991), could be fuzzified under the proposed framework". Perhaps they can, but not using the framework offered by Park and Dagher or Chung and Lee.

REFERENCES

Anderson, E. (1935). The IRISes of the Gaspe peninsula. *Bulletin of the American IRIS Society*, **59**, 2–5.

Bezdek, J. C. (1973). *Fuzzy mathematics for pattern classification.* Ph.D. Thesis, Cornell University, Ithaca, New York.

Bezdek, J. C. (1976). A physical interpretation of fuzzy ISODATA. *IEEE Transactions SMC*, **SMC-6**(5), 387–389.

Bezdek, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms.* New York: Plenum.

Bezdek, J. C. (1992a). Computing with uncertainty. *IEEE Communications Magazine*, **30**(9), 24–36.

Bezdek, J. C. (1992b). Integration and generalization of LVQ and c-means clustering, intelligent robots and computer vision Xi: biological, neural net, and 3-D models. *SPIE Proceedings 1826*, D. Cassasent, (Ed.) (pp. 280–299). Bellingham, WA: SPIE.

Bezdek, J. C., Hathaway, R. J., Howard, R. E., Wilson, C. A., & Windham, M. P. (1987). Local convergence analysis of a grouped variable version of coordinate descent. *Journal of Optimization Theory and Applications*, **54**(3), 471–477.

Bezdek, J. C., Tsao, E., & Pal, N. R. (1992). Fuzzy Kohonen clustering networks. *Proc. First IEEE Conf. on Fuzzy Systems*, (pp. 1035–1043). Piscataway, NJ: IEEE Press.

Chung, F. L., & Lee, T. (1992). Fuzzy competitive learning. *Neural Networks*, **7**(3), 539–551.

Duda, R., & Hart, P. (1973). *Pattern classification and scene analysis.* New York. John Wiley.

Dunn, J. C. (1974). A fuzzy relative of the ISODATA process and its use in detecting compact, well-separated clusters. *Journal of Cybernetics*, **3**, 32–57.

Hathaway, R. J., & Bezdek, J. C. (1991). Grouped coordinate minimization using Newton's method for inexact minimization in one vector coordinate. *Journal of Optimization Theory and Applications*, **71**(3), 503–516.

Hathaway, R. J., & Bezdek, J. C. (1995). Optimization of clustering criteria by reformulation. *IEEE Transactions on Fuzzy Systems*, **3**(2), 241–246.

Kong, S. G., & Kosko, B. (1991). Adaptive vector quantization for phoneme recognition. *IEEE Transactions on Neural Networks*, 2(1), 118–124.

Luenberger, D. L. (1965). *Introduction to linear and non-linear programming*. Reading, MA: Addison-Wesley.

MacQueen, J. (1967). Classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Math. Stat. and Prob*, pp. 281–297.

Pal, N. R., Bezdek, J. C., & Tsao, E. (1993). Generalized clustering networks and Kohonen's self organizing scheme. *IEEE Transactions on Neural Networks*, 4(4), 549–558.

Park, D. C., & Dagher, I. (1994). Gradient based fuzzy c-means (GBFCM) algorithm. *Proc. the IEEE ICNN* (Vol. 3, pp. 1626–1631). Piscataway, NJ: IEEE Press.

Redner, R., Hathaway, R. J., & Bezdek, J. C. (1987). Estimating the parameters of mixture models with modal estimators. *Commun. Stat. (A)*, 16(9), 2639–2660.

Tou, J., & Gonzalez, R. (1974). *Pattern recognition principles*. Reading, MA: Addison-Wesley.

Tsypkin, Y. Z. (1973). *Foundations of the theory of learning systems*. Trans. Z. J. Nikolic. New York: Academic Press.

Wei, W., & Mendel, J. (1994). Optimality tests for the fuzzy c-means algorithm. *Pattern Recognition*, 27(11), 1567–1573.

Windham, M. P. (1987). Parameter modification for clustering criteria. *Journal of Classification*, 4, 191–214.

APPENDIX

Let J denote J_m, $m \geq 1$, let R denote the corresponding reformulated version. Denote by M the corresponding set of partitions, i.e., M_{fcn} or M_{hcn}, and finally, let $\Phi(\mathbf{V})$ denote $(\forall i, k)$

$$\Phi(\mathbf{V}) = \begin{bmatrix} 1; & \|\mathbf{x}_k - \mathbf{v}_i\| < \|\mathbf{x}_k - \mathbf{v}_j\|,\ j \neq i \\ 0; & \text{otherwise} \end{bmatrix}$$

for the hard case, or

$$\Phi(\mathbf{V}) = \left[\sum_{j=1}^{c} \left(\frac{\|\mathbf{x}_k - \mathbf{v}_i\|_A}{\|\mathbf{x}_k - \mathbf{v}_j\|_A} \right)^{\frac{2}{m-1}} \right]^{-1}$$

for the fuzzy case.

THEOREM. Let the distances D_{ikA}, for $i = 1, \ldots, c$ and $k = 1, \ldots, n$, be continuous functions of $\mathbf{V} \in B$, where B is an open subset of $\mathscr{R}^{ca}$. Let $\mathbf{V}^*$ be such that the corresponding distances satisfying $D^*_{ikA} > 0$, for $i = 1, \ldots, c$ and $k = 1, \ldots, n$.

For either the hard or fuzzy case:

(i) $(U^*, \mathbf{V}^*)$ globally minimizes J over $M \times \mathscr{R}^{ca} \Rightarrow \mathbf{V}^*$ globally minimizes R over $\mathscr{R}^{ca}$; and
(ii) $\mathbf{V}^*$ globally minimizes R over $\mathscr{R}^{ca} \Rightarrow (\Phi(\mathbf{V}^*), \mathbf{V}^*)$ globally minimizes J over $M \times \mathscr{R}^{ca}$.

For the fuzzy case:

(iii) $(U^*, \mathbf{V}^*)$ locally minimizes $J \Rightarrow \mathbf{V}^*$ locally minimizes R; and
(iv) $\mathbf{V}^*$ locally minimizes $R \Rightarrow (\Phi(\mathbf{V}^*), \mathbf{V}^*)$ locally minimizes J.

Proof: We first do (i), using contradiction. Assume that $(U^*, \mathbf{V}^*)$ is a global minimizer of J. Since all $D^*_{ikA} > 0$, we can apply existing optimality theory for each of the cases to obtain $U^* = \Phi(\mathbf{V}^*)$, which also implies that $R(\mathbf{V}^*) = J(U^*, \mathbf{V}^*)$. Now if we also assume that $\mathbf{V}^*$ does not globally minimize R, then there is some $\mathbf{V}^{**}$ such that $R(\mathbf{V}^*) > R(\mathbf{V}^{**}) = J(\Phi(\mathbf{V}^{**}), \mathbf{V}^{**})$, but this contradicts the global optimality of $(U^*, \mathbf{V}^*)$ for J. A similar argument establishes (ii).

We now do (iii) by contradiction. Assume that $(U^*, \mathbf{V}^*)$ locally minimizes J. This implies $U^* = \Phi(\mathbf{V}^*)$, $R(\mathbf{V}^*) = J(U^*, \mathbf{V}^*)$, and the existence of an open neighborhood N in $M \times \mathscr{R}^{ca}$ that contains $(U^*, \mathbf{V}^*)$ and satisfies $J(U, \mathbf{V}) \geq J(U^*, \mathbf{V}^*)$ for every $(U, \mathbf{V}) \in N$. Now, also assume that $\mathbf{V}^*$ does not locally minimize R. This implies the existence of a sequence $\{\mathbf{V}^{(r)}\}$ in $\mathscr{R}^{ca}$, which converges to $\mathbf{V}^*$ and satisfies $R(\mathbf{V}^{(r)}) < R(\mathbf{V}^*)$ for $r = 1, \ldots$. Using the continuity (in the fuzzy case) of Φ at $\mathbf{V}^*$, this implies the existence of a sequence $\{(U^{(r)}, \mathbf{V}^{(r)})\} = \{(\Phi(\mathbf{V}^{(r)}), \mathbf{V}^{(r)})\}$, in $M \times \mathscr{R}^{ca}$, which converges to $(U^*, \mathbf{V}^*)$ and satisfies $J(U^{(r)}, \mathbf{V}^{(r)}) < J(U^*, \mathbf{V}^*)$ for $r = 1, \ldots$. This contradicts the existence of the neighborhood N described above, which contradicts the assumption that $(U^*, \mathbf{V}^*)$ locally minimizes J.

Finally, for the proof of (iv) by contradiction, we assume that $\mathbf{V}^*$ is a local minimizer of R and that $(U^*, \mathbf{V}^*) = (\Phi(\mathbf{V}^*), \mathbf{V}^*)$ is not a local minimizer of J. By the second part of this assumption, there exists a sequence $\{(U^{(r)}, \mathbf{V}^{(r)})\}$, in $M \times \mathscr{R}^{ca}$, that converges to $(U^*, \mathbf{V}^*)$ and satisfies $J(U^{(r)}, \mathbf{V}^{(r)}) < J(U^*, \mathbf{V}^*)$ for $r = 1, \ldots$. Now, $R(\mathbf{V}^{(r)}) = J(\Phi(\mathbf{V}^{(r)}), \mathbf{V}^{(r)}) \leqslant J(U^{(r)}, \mathbf{V}^{(r)}) < J(U^*, \mathbf{V}^*) = R(\mathbf{V}^*)$, for $r = 1, \ldots$. This last inequality, for the sequence $\{\mathbf{V}^{(r)}\} \to \mathbf{V}^*$, contradicts the assumption that $\mathbf{V}^*$ is a local minimizer of R. ■

Remark. A special case of the fuzzy instance of this theorem for $m > 1$ was given in Wei and Mendel (1994); and a more general case that includes the possibilistic c-means model was given in Hathaway and Bezdek (1995).