Bayesian Fuzzy Clustering

Taylor C. Glenn, *Member, IEEE*, Alina Zare, *Senior Member, IEEE*, and Paul D. Gader, *Fellow, IEEE*

Abstract—**We present a Bayesian probabilistic model and inference algorithm for fuzzy clustering that provides expanded capabilities over the traditional Fuzzy C-Means approach. Additionally, we extend the Bayesian Fuzzy Clustering model to handle a variable number of clusters and present a particle filter inference technique to estimate the model parameters including the number of clusters. We show results on synthetic and real data and compare with other approaches.**

Index Terms—**Bayes methods, clustering algorithms, clustering methods, fuzzy sets, fuzzy systems, monte carlo methods, particle filters.**

I. Introduction

FUZZY clustering is often compared with the probabilistic technique of Gaussian mixture model (GMM) estimation as the two methods do have many similarities on the surface. The superficial similarity view is where most comparisons stop, however, leaving the true differences muddled. At least partly because the two methods are seemingly similar yet somehow different, two camps arise, one of probabilistic methods for solving problems and another of fuzzy methods. The authors see merit in both approaches to solving problems and strive to assemble a toolbox consisting of the best techniques of both camps. This paper crosses between camps by presenting a probabilistic model for fuzzy clustering, and through the use of Bayesian probabilistic inference, it develops new algorithms for fuzzy clustering with expanded capabilities and improved results. Furthermore, it can be shown by this model that the fuzzy clustering problem is indeed a different problem from mixture model estimation as the two problems have different probabilistic models.

Fuzzy clustering can be viewed as the problem of finding a partitioning of the data into fuzzy sets. Each data point, $\mathbf{x}_n$ for $n = 1, \ldots, N$, has nonnegative membership, u_{nc}, in each of the fuzzy sets, indexed by $c = 1, \ldots, C$ and a total membership of one shared between all of the sets, i.e., $\sum_{c=1}^{C} u_{nc} = 1$. For example, data points representing the height values of people could be divided into two fuzzy sets. First is the set of all "tall" people and, second, the set of all "short" people. Every data point simultaneously belongs to both fuzzy sets with a varying degree of membership. A tall basketball player may have a membership value near 1 in the set of tall people and by definition the value of one minus that membership in the set of short people. In contrast with probabilistic mixture models, however, neither of these sets is considered to be an underlying generating distribution of the data point.

The goal of fuzzy clustering is to find an optimal group of sets to explain the data points, where the optimality is defined as a minimum total membership weighted distance of each data point $\mathbf{x}_n$ to a prototype of each set $\mathbf{y}_c$. Thus, the fuzzy clustering problem has two groups of unknown variables, the set prototypes $\{\mathbf{y}_c\}$ and the set memberships $\{u_{nc}\}$.

The fuzzy clustering problem has classically been tackled with the Fuzzy C-Means (FCM) algorithm [1]. FCM is derived by minimization of (1)

$$J(\mathbf{X}, \mathbf{U}, \mathbf{Y}) = \sum_{n=1}^{N} \sum_{c=1}^{C} u_{nc}^{m} \, d(\mathbf{x}_n, \mathbf{y}_c)^2 \tag{1}$$

where m is the fuzzifier parameter which affects the degree to which memberships are mixed and where the form of the distance function $d(\mathbf{x}_n, \mathbf{y}_c)$ can be used to control the type of clusters found. This is often taken to be the Euclidean distance $d(\mathbf{x}_n, \mathbf{y}_c) = \|\mathbf{x}_n - \mathbf{y}_c\|$. Values of the memberships and the cluster prototypes can be found using an alternating optimization strategy. Closed-form alternating update equations are derived through differentiation of the objective function with Lagrange multiplier constraints to yield (2) and (3)

$$u_{nc} = \frac{(1/d(\mathbf{x}_n, \mathbf{y}_c)^2)^{1/(m-1)}}{\sum_{k=1}^{C} (1/d(\mathbf{x}_n, \mathbf{y}_k)^2)^{1/(m-1)}} \tag{2}$$

$$\mathbf{y}_c = \frac{1}{\sum_{n=1}^{N} u_{nc}^{m}} \sum_{n=1}^{N} u_{nc}^{m} \mathbf{x}_n. \tag{3}$$

We note that this algorithm is guaranteed to converge (as no update can increase the objective value) but only to a local minimum or saddle point of the objective function. It is, thus, subject to the initialization conditions as to which local minimum will be found.

For the Bayesian Fuzzy Clustering (BFC) approach, we represent the unknown set memberships u_{nc} and set prototypes $\mathbf{y}_c$ as random variables. Given the specific constraints and uncertainty associated with each random variable, we select an appropriate probability distribution for each one. We then infer the most likely value of these variables given the data we have observed. BFC can then be extended to represent the number of clusters C as a random variable as well, yielding a new approach to estimating this quantity.

A commonly cited problem with traditional clustering approaches is that the number of clusters must be specified as a parameter to the algorithm. However, its value is often unknown to the user of the algorithm. In the fuzzy clustering literature, one approach often taken is to use each value within a reasonable range and to compare the results using a cluster

Manuscript received January 28, 2014; revised June 1, 2014; accepted September 19, 2014. Date of publication November 13, 2014; date of current version October 2, 2015.

T. C. Glenn and P. D. Gader are with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: tcg@cise.ufl.edu; pgader@cise.ufl.edu).

A. Zare is with the Department of Electrical and Computer Engineering, University of Missouri, Columbia, MO 65211 USA (e-mail: zarea@missouri.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2014.2370676

validity index. Such an approach was taken in an influential early work by Gath and Geva [2]. In contrast, a fuzzy clustering algorithm that attempts to learn the number of clusters is Competitive Agglomeration (CA) [3]. This algorithm starts with a higher than expected number of clusters and places them in competition against each other to receive the membership of the data points. As the algorithm runs, it progressively prunes clusters which have low total membership. Another fuzzy clustering approach presented by Li *et al.* [4] also starts with a large number of clusters but allows cluster prototypes to become colocated. The colocated clusters are then merged in a postprocessing step after the algorithm converges.

Due to the prevalence of both fuzzy and probabilistic techniques, others have also used them in some combination. In particular, a prior approach was taken by Theis [5] that used Bayesian techniques to estimate a fuzzy clustering of colored graphs; however, this work does not address the general fuzzy clustering problem. Other works have also tangentially combined fuzzy clustering with Bayesian techniques [6]–[8] or used fuzzy and Bayesian methods in the same system [9]–[15]. We have not, however, seen prior work that directly approaches the fuzzy clustering problem using Bayesian probabilistic modeling and inference.

In the following sections, we derive the BFC model, present a Markov-Chain Monte-Carlo (MCMC) method for inference and compare the results of this model with traditional FCM clustering. We then extend the model to estimate the number of clusters in the data, provide an efficient method for inference in this model, show its results on real data, and compare the results of this algorithm with other fuzzy clustering approaches which estimate the number of clusters.

The presented models and algorithms can replicate and extend the results obtained by FCM. For example, the "fuzzifier" m parameter is no longer constrained to be greater than 1. Theoretically, given that the proposed approach makes use of MCMC sampling techniques, the global optima (as opposed to local optima found with the standard FCM and k-means algorithms) can be estimated. In addition, the proposed clustering approaches can be extended for use with distance measures in which there are no closed-form updates for the parameter values [16]. The approach for estimating the number of clusters can also be adapted to any application that uses an FCM-like objective.

II. Bayesian Fuzzy Clustering Model

The BFC model is composed of a data likelihood distribution (4) called the Fuzzy Data Likelihood (FDL)

$$p(\mathbf{X}|\mathbf{U},\mathbf{Y}) = \prod_{n=1}^{N} \mathrm{FDL}(\mathbf{x}_n|\mathbf{u}_n,\mathbf{Y})$$

$$= \prod_{n=1}^{N} \frac{1}{Z(\mathbf{u}_n, m, \mathbf{Y})} \prod_{c=1}^{C} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu} = \mathbf{y}_c, \boldsymbol{\Lambda} = u_{nc}^m\,\mathbf{I}) \quad (4)$$

a prior distribution for the cluster memberships (5) called the Fuzzy Cluster Prior (FCP)

$$\tilde{p}(\mathbf{U}|\mathbf{Y}) = \prod_{n=1}^{N} \mathrm{FCP}(\mathbf{u}_n|\mathbf{Y})$$

$$= \prod_{n=1}^{N} Z(\mathbf{u}_n, m, \mathbf{Y}) \left(\prod_{c=1}^{C} u_{nc}^{-mD/2}\right) \mathrm{Dirichlet}(\mathbf{u}_n|\boldsymbol{\alpha}) \quad (5)$$

and a Gaussian prior distribution on cluster prototypes (6)

$$p(\mathbf{Y}) = \prod_{c=1}^{C} \mathcal{N}(\mathbf{y}_c|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y) \quad (6)$$

where N is the number of data points, C is the number of clusters, D is the dimensionality of the data, u_{nc} is the membership of data point $\mathbf{x}_n$ in cluster c, m is the fuzzifier, $\mathbf{y}_c$ are the cluster prototypes, $\mathbf{I}$ is the D dimensional identity matrix, and $Z(\mathbf{u}_n, m, \mathbf{Y})$ is a normalization constant given in (7). The arguments to the probability density functions are grouped into the $D \times N$ matrix of data points $\mathbf{X} = [\mathbf{x}_1|\ldots|\mathbf{x}_N]$, memberships $\mathbf{U}$ with dimensions $C \times N$ where the element at row c, column n is u_{nc}, and prototypes $\mathbf{Y} = [\mathbf{y}_1|\ldots|\mathbf{y}_C]$ with dimensions $D \times C$.

In the FDL (4), data are given a likelihood proportional to the product of C normal likelihoods, each with a different precision. This might be viewed as the likelihood of the C distributions simultaneously generating the data point. The precision of each of the normal components $\lambda = u_{nc}^m$ is specific to each data point. Thus, the far away normal components of the likelihood are stretched-out by using a low membership/precision (high variance) value. Note that, in this model, each data point has a unique vector of parameters for the data likelihood. Therefore, each data point can be considered to have its own generating probability distribution. Across all data points, however, the normal likelihoods are grouped such that they share mean values $\boldsymbol{\mu} = \mathbf{y}_c$ which are the cluster prototypes.

We note that the product of Gaussian likelihood functions over the same variable $\mathbf{x}_n$ is no longer a normalized distribution, although the new normalization constant is computable as

$$Z(\mathbf{u}_n, m, \mathbf{Y}) = (2\pi)^{-\frac{D}{2}(C-1)} \left(\prod_{c=1}^{C} u_{nc}^m\right)^{\frac{D}{2}} \left(\sum_{c=1}^{C} u_{nc}^m\right)^{-\frac{D}{2}}$$
$$\times \exp\left\{-\frac{1}{2}\left(\sum_{c=1}^{C} u_{nc}^m \mathbf{y}_c^T \mathbf{y}_c - \frac{||\sum_{c=1}^{C} u_{nc}^m \mathbf{y}_c||^2}{\sum_{c=1}^{C} u_{nc}^m}\right)\right\} \quad (7)$$

where the derivation of this function is given as an appendix. However, in the inference algorithms discussed in this paper, the normalization constant is canceled by the FCP and, thus, need not be computed. We note also that the product of normals in the FDL can be rewritten as a single normal distribution for which the mean and covariance can be computed by expanding terms and completing the square. This single normal version shows how to derive the FDL normalization constant. We prefer the product of normals notation as it more clearly expresses the intent of the model and more directly sets up the inference algorithms.

The membership variables are assumed to have a prior distribution as shown in (5). We call this prior, which we derived for this purpose, the FCP. This prior consists of three factors which we will call $F_1 = Z(\mathbf{u}_n, m, \mathbf{Y}), F_2 = \left(\prod_{c=1}^{C} u_{nc}^{-mD/2}\right)$, and $F_3 = \mathrm{Dirichlet}(\mathbf{u}_n|\boldsymbol{\alpha})$. The first two factors are a counterbalance to the data likelihood with F_1 exactly canceling the normalization constant of the FDL.

The factor F_2 occurs because the Gaussian components of (4) will promote high membership values, having the factor of $u_{nc}^{mD/2}$ which is large for large memberships. The individual factors of F_2 (each $u_{nc}^{-mD/2}$), though, are largest for small membership values. The factors cancel, causing the joint distribution to be more agnostic about membership values. Because of the negative value of the exponent in F_2, however, the FCP cannot be normalized over the interval $[0, 1]$ (at least for many relevant values of m and D). Thus, it belongs to the commonly used class of improper priors. Although improper priors are more typically used to provide a distribution which is uninformative about prior belief, the FCP is used as a mathematical convenience to exactly replicate the behavior of FCM by the Bayesian model. We note that a proper prior that is a product of Inverse-Gamma distributions could be used in place of F_2 in the FCP. If the shape parameter of the Inverse-Gamma is set to $\alpha = mD/2 - 1$ and scale parameter β is small, then the performance of the model would be unchanged.

The F_3 factor is a Dirichlet likelihood parameterized by the vector $\boldsymbol{\alpha}$, defined as

$$\text{Dirichlet}(\mathbf{x}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{k=1}^{K} \alpha_k)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} x_k^{\alpha_k - 1} \tag{8}$$

on the domain of the standard simplex, $x_k \geq 0$ for all $k = 1, \ldots, K, \sum_{k=1}^{K} x_k = 1$. When taken to be a symmetric Dirichlet distribution with parameter $\boldsymbol{\alpha} = \mathbf{1}_C$, that is a $C \times 1$ column vector of all ones, this factor vanishes and has no effect on the clustering output. Thus, this factor is not strictly necessary to replicate the behavior of the FCM algorithm. The addition of the Dirichlet factor, however, nicely expresses the positivity and sum-to-one constraints on the memberships, and it provides additional flexibility and capability to the clustering algorithm while containing FCM as a subcase of the more general method.

Finally, the BFC model places a Gaussian prior (6) on each of the cluster prototype parameters. The hyperparameters of this distribution can be set in the empirical Bayes fashion to use the mean of the dataset

$$\boldsymbol{\mu}_y = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n \tag{9}$$

and a wide covariance in the shape of the data covariance

$$\Sigma_y = \frac{\gamma}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu}_y)(\mathbf{x}_n - \boldsymbol{\mu}_y)^T \tag{10}$$

where γ is a user set parameter affecting the strength of the prior, we used $\gamma = 3$ in our applications. Considering the log-likelihood of the full BFC model as a fuzzy clustering objective function, the cluster prototype prior acts as a regularization term.

When multiplied together, the FDL (4), FCP (5), and prototype prior (6) form the joint likelihood of the data and parameters

$$\begin{aligned} p(\mathbf{X}, \mathbf{U}, \mathbf{Y}) &= p(\mathbf{X}|\mathbf{U}, \mathbf{Y})\tilde{p}(\mathbf{U}|\mathbf{Y})p(\mathbf{Y}) \\ &\propto \exp\left\{-\frac{1}{2}\sum_{n=1}^{N}\sum_{c=1}^{C} u_{nc}^{m}\|\mathbf{x}_n - \mathbf{y}_c\|^2\right\}\left[\prod_{n=1}^{N}\prod_{c=1}^{C} u_{nc}^{\alpha_c - 1}\right. \\ &\quad \left. \times \exp\left\{-\frac{1}{2}\sum_{c=1}^{C}(\mathbf{y}_c - \mu_y)^T \boldsymbol{\Sigma}_y^{-1}(\mathbf{y}_c - \mu_y)\right\}\right] \end{aligned} \tag{11}$$

which is proportional to the posterior distribution of the parameters, $p(\mathbf{U}, \mathbf{Y}|\mathbf{X}) \propto p(\mathbf{X}, \mathbf{U}, \mathbf{Y})$. The objective function form of the joint likelihood is the negative of its logarithm

$$\begin{aligned} J(\mathbf{X}, \mathbf{U}, \mathbf{Y}) = &\sum_{n=1}^{N}\sum_{c=1}^{C} u_{nc}^{m}\|\mathbf{x}_n - \mathbf{y}_c\|^2 \\ &- 2\sum_{n=1}^{N}\sum_{c=1}^{C}(\alpha_c - 1)\log(u_{nc}) \\ &+ \sum_{c=1}^{C}(\mathbf{y}_c - \mu_y)^T \boldsymbol{\Sigma}_y^{-1}(\mathbf{y}_c - \mu_y) \end{aligned} \tag{12}$$

where a factor of 2 has been multiplied through to simplify.

As an important point of discussion, comparison of the BFC model with the probabilistic GMM often used for clustering is needed. The complete-data likelihood for the GMM is shown as follows:

$$p(\mathbf{X}|\mathbf{Z}, \mathbf{Y}) = \prod_{n=1}^{N}\prod_{c=1}^{C} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)^{z_{nc}} \tag{13}$$

where $z_{nc} \in \{0, 1\}$, and $\sum_{c=1}^{C} z_{nc} = 1$.

The mixture model has a fundamentally different interpretation than the proposed BFC model. The GMM has the underlying assumption that each data point is a sample generated from only *one* "true" generating component of the C mixture components. This is expressed by the (latent) variables z_{nc} which, for each data point $\mathbf{x}_n$, uses one of the mixture component likelihoods and raises the rest to the power zero, eliminating them from the model. In contrast, the BFC approach follows assumptions from the fuzzy clustering literature in which each data point has partial membership in all clusters. In other words, no specific cluster is assumed to be the "true" generating cluster for each data point. Comparing the likelihood in (13) to the FDL (4), we can see that the forms of the two likelihoods are significantly distinct. In particular, rather than a selection of a single "true" generating mixture component, a product of all component likelihoods is used. Furthermore, the FDL is data point specific, since the inverse-covariance for each of the component distributions is weighted by the data point-specific memberships u_{nc}^{m}. This results in the FDL being independently but *not* identically distributed over the data points. In contrast, mixture model approaches assume independence while being identically distributed over the data points. The GMM can be extended to incorporate prior distributions over the parameters of interest, and Bayesian parameter estimation approaches can be employed to estimate the associated means and covariances for a GMM. However, as the data likelihood between the GMM and the FDL are distinct, extension of the GMM with the inclusion of prior distributions would not produce a model similar to the BFC models proposed here.

Algorithm 1: Bayesian Fuzzy Clustering MAP Search

```
Data: Data matrix $\mathbf{X}$, fuzzifier $m$, number of clusters $C$,
      number of sampling iterations $N_{\text{iter}}$
Result: MAP estimates for membership $\mathbf{U}^*$ and cluster
        prototypes $\mathbf{Y}^*$
initialize hyperparameters $\mu_y, \mathbf{\Sigma}_y$ from (9) and (10)
sample initial $\mathbf{u}_n \sim \text{Dirichlet}(\boldsymbol{\alpha} = \mathbf{1}_C)$ for all
  $n = 1 \ldots N$
sample initial $\mathbf{y}_c \sim \mathcal{N}(\boldsymbol{\mu}_y, \mathbf{\Sigma}_y)$ for all $c = 1 \ldots C$
set MAP samples to current $\mathbf{u}_n^* \leftarrow \mathbf{u}_n, \mathbf{y}_c^* \leftarrow \mathbf{y}_c$ for all
  $n = 1 \ldots N, c = 1 \ldots C$
for iter $= 1 \ldots N_{\text{iter}}$ do
    /* sample $\mathbf{U} \sim p(\mathbf{U}|\mathbf{X},\mathbf{Y}) \propto p(\mathbf{X},\mathbf{U},\mathbf{Y})$ */
    for $n = 1 \ldots N$ do
        sample proposed new membership vector $\mathbf{u}_n^\dagger$
          from (14)
        accept proposal ($\mathbf{u}_n \leftarrow \mathbf{u}_n^\dagger$) with probability $a_{\mathbf{u}}$
          from (16)
        if $p(\mathbf{x}_n, \mathbf{u}_n^\dagger|\mathbf{Y}^*) > p(\mathbf{x}_n, \mathbf{u}_n^*|\mathbf{Y}^*)$, using (15) then
            $\mathbf{u}_n^* \leftarrow \mathbf{u}_n^\dagger$
    /* sample $\mathbf{Y} \sim p(\mathbf{Y}|\mathbf{X},\mathbf{U}) \propto p(\mathbf{X},\mathbf{U},\mathbf{Y})$ */
    for $c = 1 \ldots C$ do
        sample proposed new cluster prototype $\mathbf{y}_c^\dagger$ from
          (17)
        accept proposal ($\mathbf{y}_c \leftarrow \mathbf{y}_c^\dagger$) with probability $a_{\mathbf{y}}$
          from (19)
        if $p(\mathbf{X}, \mathbf{y}_c^\dagger|\mathbf{U}^*) > p(\mathbf{X}, \mathbf{y}_c^*|\mathbf{U}^*)$, using (18) then
            $\mathbf{y}_c^* \leftarrow \mathbf{y}_c^\dagger$
    /* check full sample for new maximum
       likelihood */
    if $p(\mathbf{X},\mathbf{U},\mathbf{Y}) > p(\mathbf{X},\mathbf{U}^*,\mathbf{Y}^*)$, using (11) then
        $\mathbf{U}^* \leftarrow \mathbf{U}$
        $\mathbf{Y}^* \leftarrow \mathbf{Y}$
```

III. Inference for Bayesian Fuzzy Clustering

The primary inference task we are concerned with is finding the maximum-*a-posteriori* (MAP) values of the parameters in the BFC model given the data. This is equivalent to finding the globally optimal values of the parameters for a regularized FCM objective function. To perform the MAP inference, we use an MCMC technique to leverage its optimality guarantees.

A Metropolis within Gibbs [17] sampler can be used to generate samples from the BFC model's posterior distribution. As the samples are generated, they can be evaluated in the posterior and the best sample is retained. Algorithm 1 gives the procedure for this method.

Sampling from the conditional distribution of the memberships given the data and cluster prototypes $p(\mathbf{U}|\mathbf{X},\mathbf{Y})$ is accomplished with a Metropolis-Hastings sampling step using, for simplicity, a uniform symmetric Dirichlet proposal distribution,

$$\mathbf{u}_n^\dagger \sim \text{Dirichlet}(\boldsymbol{\alpha} = \mathbf{1}_C). \tag{14}$$

The conditional distribution of the memberships $p(\mathbf{U}|\mathbf{X},\mathbf{Y})$ is proportional to the joint distribution of data, memberships, and prototypes, $p(\mathbf{X},\mathbf{U},\mathbf{Y})$, given fixed values of the cluster prototypes. Additionally, for any proposed membership vector $\mathbf{u}_n^\dagger$ for a data point index n, the terms related to the other membership vectors and cluster prototypes are unchanged. Therefore, we only need to evaluate the following quantity:

$$\begin{aligned}\tilde{p}(\mathbf{x}_n, \mathbf{u}_n|\mathbf{Y}) &= p(\mathbf{x}_n|\mathbf{u}_n,\mathbf{Y})\tilde{p}(\mathbf{u}_n|\mathbf{Y}) \\ &= \prod_{c=1}^{C} \mathcal{N}(\mathbf{x}_n|\mathbf{y}_c, u_{nc}^m)\, u_{nc}^{-mD/2}\text{Dirichlet}(\mathbf{u}_n|\boldsymbol{\alpha}) \\ &\propto \prod_{c=1}^{C} \exp\left\{-\frac{1}{2}u_{nc}^m\|\mathbf{x}_n - \mathbf{y}_c\|^2\right\} u_{nc}^{\alpha_c - 1}. \end{aligned}\tag{15}$$

A proposed new membership sample $\mathbf{u}_n^\dagger$ is, thus, accepted to replace $\mathbf{u}_n$ with probability equal to the ratio

$$a_{\mathbf{u}} = \min\left\{1, \frac{\tilde{p}(\mathbf{x}_n, \mathbf{u}_n^\dagger|\mathbf{Y})}{\tilde{p}(\mathbf{x}_n, \mathbf{u}_n|\mathbf{Y})}\right\} \tag{16}$$

where as the proposal distribution is not dependent upon the current sample, a Hastings correction is not needed.

The next Gibbs sampling step is to sample new cluster prototypes from the conditional distribution of the prototypes given the data and memberships $p(\mathbf{Y}|\mathbf{X},\mathbf{U})$. This distribution is proportional to the joint distribution $p(\mathbf{X},\mathbf{U},\mathbf{Y})$ for fixed values of the data and memberships. Proposed new values for the cluster prototypes are sampled from a Gaussian distribution with small variance in the shape of the prior and centered on the current Markov-chain state

$$\mathbf{y}_c^\dagger \sim \mathcal{N}\left(\mathbf{y}_c, \frac{1}{\delta}\mathbf{\Sigma}_y\right) \tag{17}$$

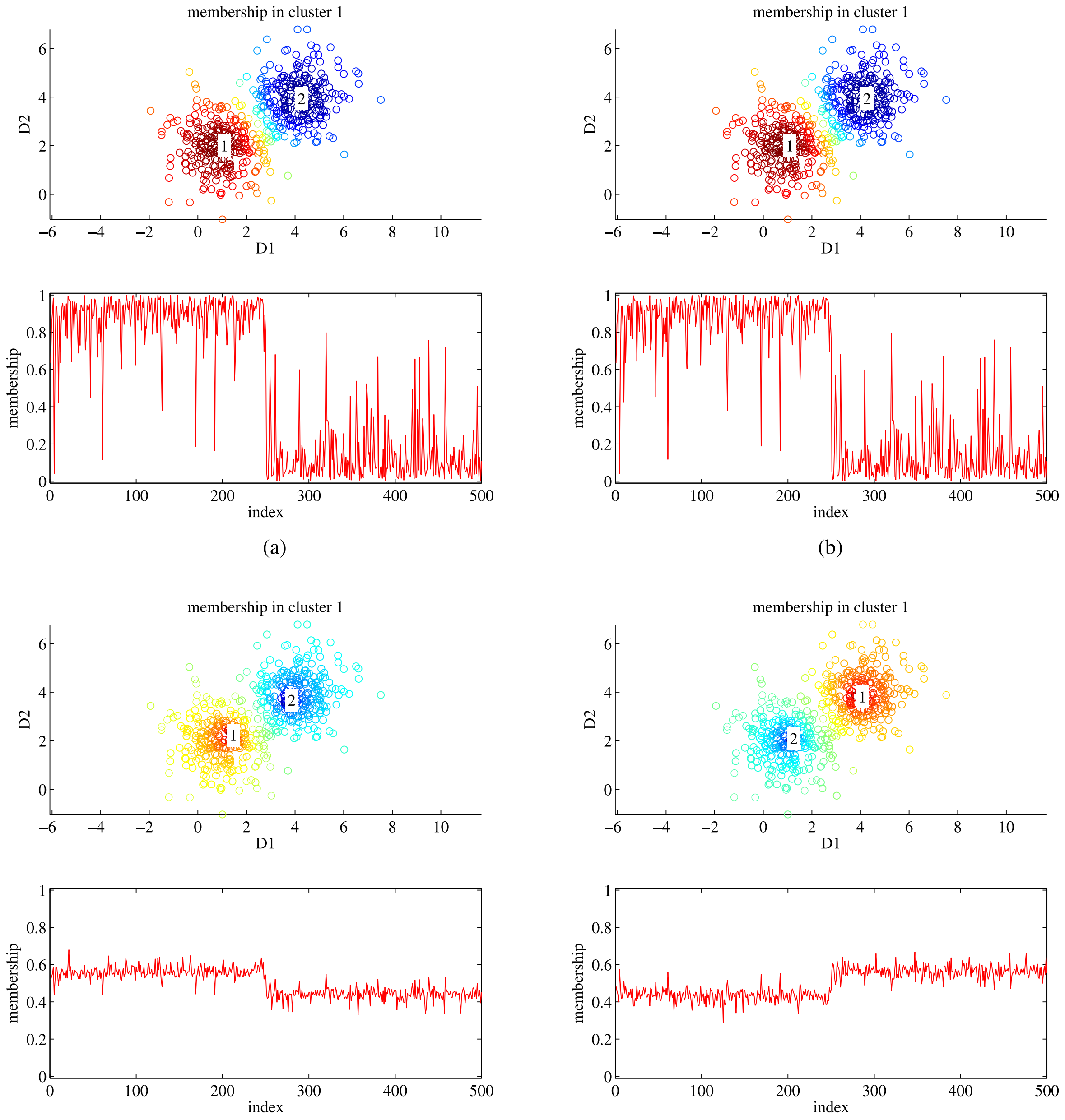
where δ is a user set parameter that controls tightness of the proposal around the current state and relates to the sample acceptance rate. In application, we set $\delta = 10$. For a single proposed cluster prototype $\mathbf{y}_c^\dagger$ the terms depending upon the other cluster prototypes and memberships will be unchanged, leaving only the following quantity to be evaluated:

$$\begin{aligned}p(\mathbf{X}, \mathbf{y}_c|\mathbf{U}) &= p(\mathbf{X}|\mathbf{U}, \mathbf{y}_c)p(\mathbf{y}_c) \\ &\propto \exp\left\{-\frac{1}{2}\sum_{n=1}^{N} u_{nc}^m\|\mathbf{x}_n - \mathbf{y}_c\|^2\right\} \\ &\quad \times \exp\left\{-\frac{1}{2}(\mathbf{y}_c - \mu_y)^T\mathbf{\Sigma}_y^{-1}(\mathbf{y}_c - \mu_y)\right\}. \end{aligned}\tag{18}$$

This gives that the proposed sample can be accepted according to the ratio

$$a_{\mathbf{y}} = \min\left\{1, \frac{p(\mathbf{X}, \mathbf{y}_c^\dagger|\mathbf{U})}{p(\mathbf{X}, \mathbf{y}_c|\mathbf{U})}\right\} \tag{19}$$

where as the Gaussian proposal is symmetric, a Hastings correction is again not needed.



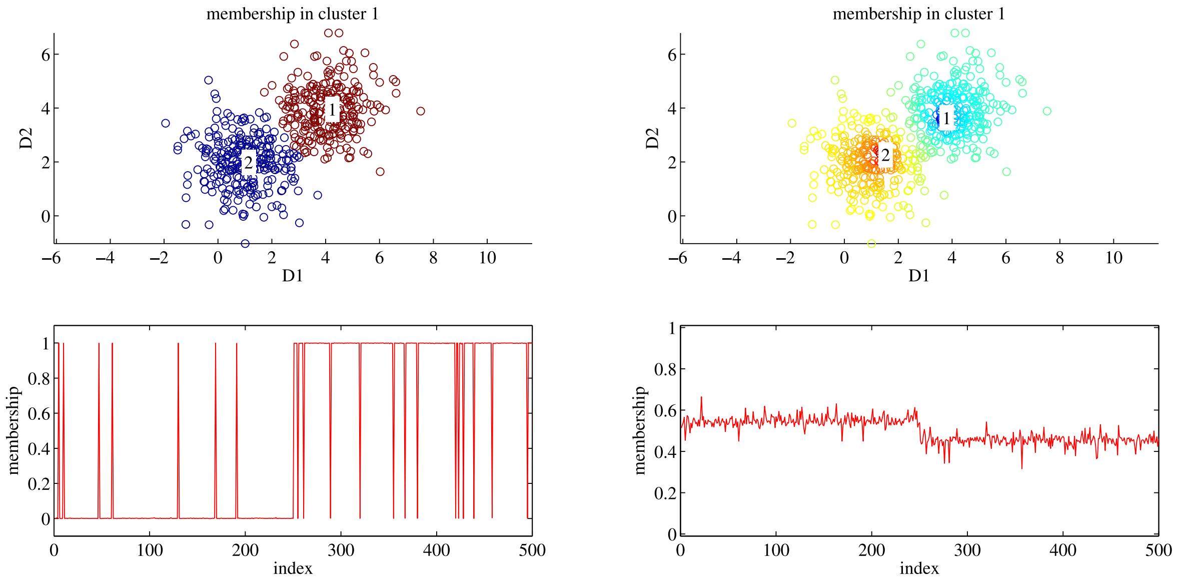
(a) (b) (c) (d)

Fig. 1. BFC and FCM outputs for $m = 2$. Scatter plots are color coded by membership in cluster 1 (red is high membership, blue is low membership). Line plots denote membership in cluster 1 by data point index. Indexes 1–250 generated by $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$, indexes 251–500 generated by $\mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$. (a) BFC MAP search results $m = 2$, $\boldsymbol{\alpha} = \mathbf{1}$. (b) FCM results $m = 2$. (c) BFC results for $m = 10$, $\boldsymbol{\alpha} = \mathbf{1}$. (d) FCM results for $m = 10$.

The search compares each proposed vector of membership values $\mathbf{u}_n^\dagger$ (whether or not it is accepted as the new Markov-chain state $\mathbf{u}_n$) for improvement over $\mathbf{u}_n^*$ in the current MAP sample $\{\mathbf{U}^*, \mathbf{Y}^*\}$ using (15). If $\mathbf{u}_n^\dagger$ increases the likelihood of the MAP sample, then it becomes the new $\mathbf{u}_n^*$. Similarly, each proposed cluster prototype $\mathbf{y}_c^\dagger$ is tested for improvement using (18) and retained as the new $\mathbf{y}_c^*$ if it is an improvement. After each full sample $\{\mathbf{U}, \mathbf{Y}\}$ is generated, its likelihood is compared with the current MAP sample $\{\mathbf{U}^*, \mathbf{Y}^*\}$ using $p(\mathbf{X}, \mathbf{U}, \mathbf{Y})$, and is retained as the new MAP sample if it has higher likelihood. We refer to the comparison of proposal samples with MAP samples as a ratcheting search and find that it speeds the search for a good answer early on, while leaving open the possibility to find a new MAP full sample as the sampler progresses.

The asymptotic computational complexity of a single iteration of this algorithm is $O(NCD + CD^2)$, where N is the number of data points, C is the number of clusters, and D is the dimensionality of the data. The quadratic dependence on D comes from the use of a full covariance matrix in the cluster prototype prior (6). An implementation restricted to use a diagonal covariance will scale at $O(NCD)$.

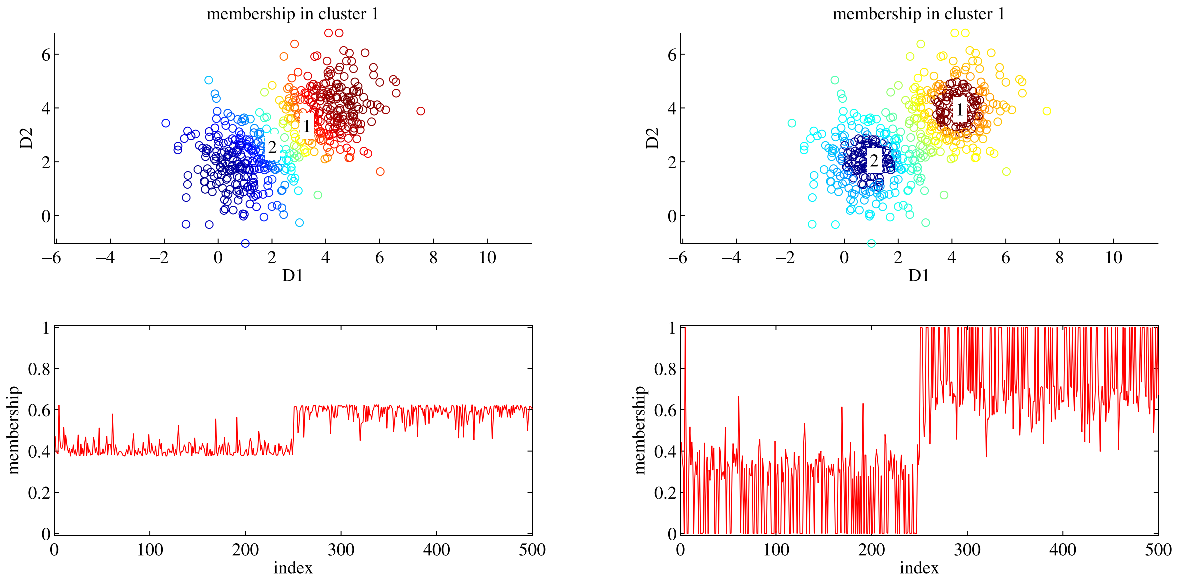
The MCMC sampler approach used in the BFC MAP search is guaranteed to converge to a global optimum given enough samples. However, the rate and determination of convergence is



(a) BFC $m = 1, \boldsymbol{\alpha} = 1$

(b) BFC $m = -10, \boldsymbol{\alpha} = 1$

Fig. 2. BFC results for $m = 1$ and $m = -10$. (a) Fuzzifier $m = 1$ results in a crisp partitioning all memberships 0 or 1. (b) Fuzzifier $m = -10$ places cluster membership away from the prototype location.



(a) BFC MAP Search $m = 2, \boldsymbol{\alpha} = 3$

(b) BFC MAP Search $m = 4, \boldsymbol{\alpha} = 0.95$

Fig. 3. BFC results for combinations of m and $\boldsymbol{\alpha}$. (a) Results for a low fuzzifier $m = 2$, and high Dirichlet concentration $\boldsymbol{\alpha} = 4$, memberships increase with distance from cluster boundary. (b) Results showing a binary inner core with high fuzzifier $m = 4$ and low Dirichlet concentration $\boldsymbol{\alpha} = 0.95$.

not predictable. It will depend upon the structure of the dataset and the number of parameters to be estimated, as well as parameter settings for the model and proposal distributions.

We also note that in this inference method, because we are not using the closed-form FCM update equations (2) and (3), we are free to set the value of the fuzzifier parameter m to 1 to obtain a crisp clustering, or even to values less than one or negative to obtain some interesting results. Furthermore, the methods can be applied with distance metrics that would not yield closed-form updates in traditional FCM.

A. Parameter Settings

The BFC MAP search algorithm has four parameters that must be set prior to running. These parameters are listed below with a discussion of how we have been setting them in our implementation and experiments.

1) *Fuzzifier—m:* The m parameter plays the same role as the fuzzifier in the standard FCM algorithm. Thus, as m approaches 1, membership values become more crisp. In most of our experiments, we set m to either 1.5 (for more crisp results) or 2 (for more fuzzy results). However, as

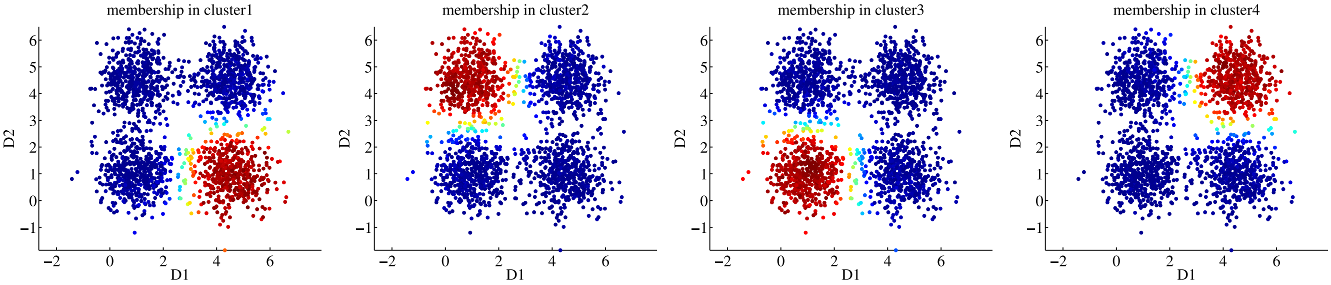


Fig. 4. FCM-like BFC membership outputs with 1000 iterations $m = 1.5$ and $\boldsymbol{\alpha} = 1$ on four-cluster Gaussian data.

with the standard FCM algorithm, setting the fuzzifier value is dependent on the particular dataset and application.

2) *Number of Clusters:* The number of clusters parameter is a key parameter in the BFC model. As with any clustering algorithm where the number of clusters must be set, this parameter is data and application dependent.

3) *Dirichlet Prior Parameter—α:* The α parameter controls the shape of the Dirichlet prior on the membership values and should be set greater than zero. When α is less than one, membership values that tend toward binary values (0 or 1) are preferred. When alpha is greater than one, fuzzy membership values are preferred. As alpha increases, the variance of the Dirichlet distribution centered on the mean membership value decreases. For experiments in which the desired membership distribution is unknown, α should be set to 1 such that the prior on the membership values is uniform (i.e., uninformative).

4) *Prototype Prior Variance Scale—γ:* The γ parameter controls the scale of the covariance of Gaussian prior on cluster prototypes in the suggested empirical approach (10). Small values (e.g., $\gamma < 1$) will encourage prototypes to be near the dataset mean, while large values will reduce the prior's influence on the result. In implementation, we have set this value to 3.

IV. Experiments and Comparisons With Fuzzy C-Means

To demonstrate the BFC sampler, two synthetic 2-D datasets were generated, one with two Gaussian clusters and one with four Gaussian clusters. For the dataset with two Gaussian clusters, 250 data points were generated for each cluster. The data points were sampled from $\boldsymbol{\mu}_1 = (1\ \ 2)T$, $\boldsymbol{\Sigma}_1 = \mathbf{I}$, and $\boldsymbol{\mu}_2 = (4\ \ 4)T$, $\boldsymbol{\Sigma}_2 = \mathbf{I}$. In the experiments for the two-cluster data, FCM was run to convergence, and the BFC search was run for 1000 sample iterations. We show results on this simple dataset to give some intuition of the behavior of the BFC model at various parameter settings. We also demonstrate that because the BFC search does not use the closed-form update equations, it can be run for settings of parameters not normally feasible for FCM.

Fig. 1 shows the results of both traditional FCM and the BFC MAP search for the fuzzifier set at $m = 2$ and $m = 10$, respectively. These results demonstrate that the ratcheting BFC search, in relatively few sampling iterations, can find a comparable solution to FCM.

Where the first experiments show that BFC and FCM produce similar results, the experiments shown in Fig. 2 give the BFC output for fuzzifier settings not normally considered valid for FCM implementations. In these experiments, the fuzzifier was set to $m = 1$ and $m = -10$, and the Dirichlet concentration parameter to $\boldsymbol{\alpha} = \mathbf{1}$. For $m = 1$, the BFC produces a crisp partitioning similar to that of K-means clustering. While for the negative value $m = -10$ the cluster prototypes are in similar locations, but high membership values are found opposite of the prototype locations. These opposite memberships are not the same as one minus the membership in a class, because in the case of more than two clusters they are still constrained to sum to 1 over all components.

Changing the Dirichlet concentration parameter provides some clustering behaviors not possible with traditional FCM clustering. Fig. 3(a) shows results using $m = 2$ and a strong Dirichlet concentration at $\boldsymbol{\alpha} = 3$. Here, the results appear spatially similar to a linear classifier output. The memberships increase to their maximum (around 0.6) as distance from the cluster border increases, whereas traditional FCM memberships tend to fall-off radially away from the cluster prototype. Contrasting the character of this distribution in the line plot to the line plots of Fig. 1(c) and (d), this distribution is more binarized within the envelope of the 0.6–0.4 membership range.

Fig. 3(b) shows another interesting behavior when the value of the fuzzifier is relatively high, at $m = 4$, while the Dirichlet concentration is less than one, at $\boldsymbol{\alpha} = 0.95$. This parameter combination shows an inner core nearest the cluster prototypes where cluster membership is crisp, while beyond this region the memberships are more mixed.

In order to illustrate that the proposed algorithm can be run on data with a larger number of clusters, four Gaussian cluster data were also generated. Two hundred data points were generated for each cluster. The data points were sampled from Gaussians with $\boldsymbol{\mu}_1 = (4.5\ \ 1)^T$, $\boldsymbol{\Sigma}_1 = 0.5 \times \mathbf{I}$, $\boldsymbol{\mu}_2 = (1\ \ 4.5)^T$, $\boldsymbol{\Sigma}_2 = 0.5 \times \mathbf{I}$, $\boldsymbol{\mu}_3 = (1\ \ 1)^T$, $\boldsymbol{\Sigma}_3 = 0.5 \times \mathbf{I}$, and $\boldsymbol{\mu}_4 = (4.5\ \ 4.5)^T$, $\boldsymbol{\Sigma}_4 = 0.5 \times \mathbf{I}$. Four experiments were run with the four Gaussian cluster data. The first experiment was run for 1000 iterations with $m = 1.5$ and $\boldsymbol{\alpha} = 1$. This experiments illustrates that the BFC algorithm can produce results similar to FCM on the four-cluster data. The results from this experiment are shown in Fig. 4. The second experiment, with results shown in Fig. 5, illustrates the ability to generate crisp

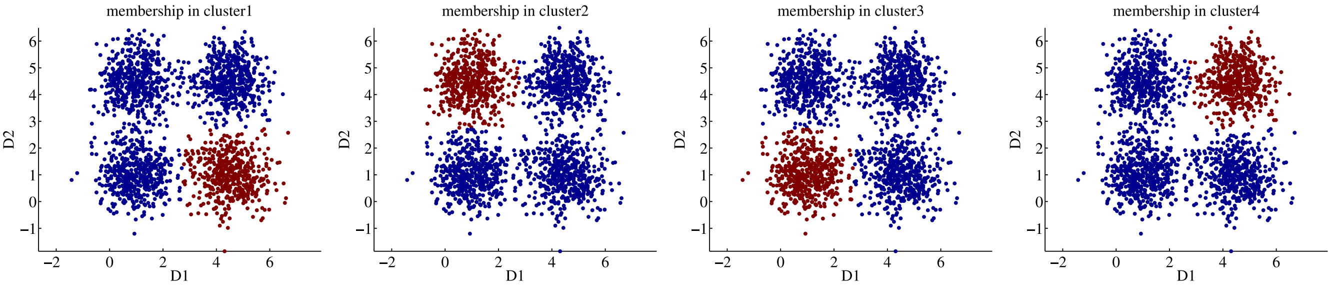


Fig. 5. Crisp BFC membership outputs with 15 000 iterations $m = 1$ and $\boldsymbol{\alpha} = 1$ on four-cluster Gaussian data.

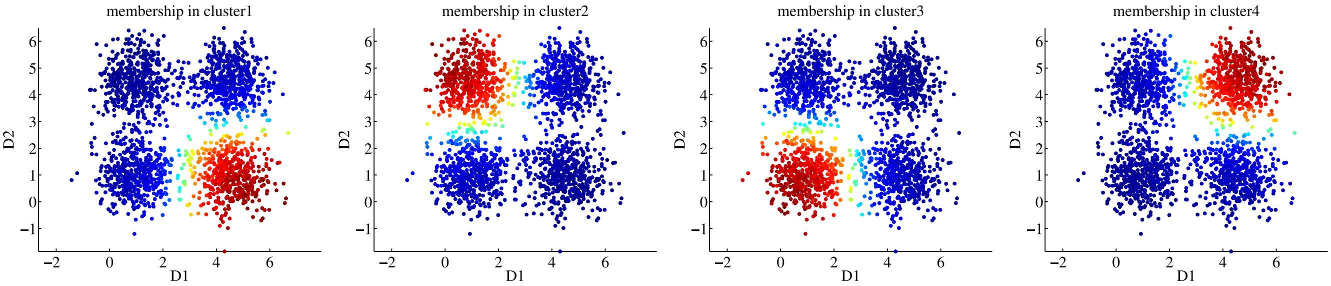


Fig. 6. Classifier-like BFC membership outputs with 5000 iterations $m = 1.2$ and $\boldsymbol{\alpha} = 1.3$ on four-cluster Gaussian data.

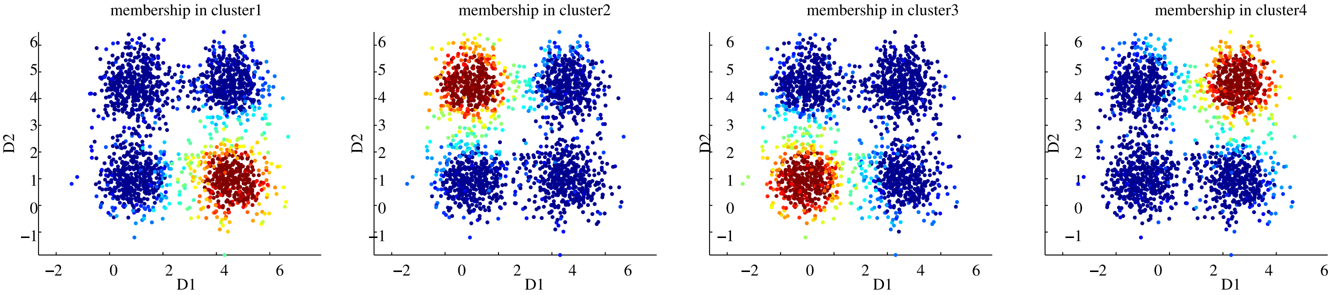


Fig. 7. Binary inner-core BFC membership outputs with 1000 iterations $m = 2.7$ and $\boldsymbol{\alpha} = 0.99$ on four-cluster Gaussian data.

membership outputs with the parameter $m = 1$. The ability to generate "classifier-like" outputs with four-cluster Gaussian data is shown in Fig. 6. These results were generated with 5000 iterations, $m = 1.2$ and $\boldsymbol{\alpha} = 1.3$. Finally, the fourth experiment illustrates the ability to generate a "binary inner-core" with the four-cluster Gaussian data. These results are shown in Fig. 7 and were obtained with 1000 iterations, $m = 2.7$ and $\boldsymbol{\alpha} = .99$.

V. Bayesian Model for Estimating the Number of Clusters

Comparing the objective value of two models of different size is not generally straightforward as the value is dependent upon the model size. This is because the additional terms in a larger model tend to increase the value regardless of whether or not the larger model is a better solution. In our approach to estimating model size, we make the assumption that taking the average of the objective value over the number of model components will remove some dependence on model size and then use a size penalty to discriminate against larger models that are otherwise equally good. For example, if an objective function is structured as $J(\mathbf{X}, \Theta) = \sum_{c=1}^{C} f(\mathbf{X}, \boldsymbol{\theta}_c)$, we restructure this as the averaged objective plus a penalty, $J'(\mathbf{X}, \Theta, C) = \frac{1}{C}\sum_{c=1}^{C} f(\mathbf{X}, \boldsymbol{\theta}_c) + \text{Penalty}(C)$.

Applying this to the BFC objective (12), we can derive a new objective averaged over model size

$$\begin{aligned} J(\mathbf{X}, \mathbf{U}, \mathbf{Y}, \mathbf{C}) &= \frac{1}{C}\sum_{n=1}^{N}\sum_{c=1}^{C} u_{nc}^{m}\|\mathbf{x}_n - \mathbf{y}_c\|^2 \\ &- \frac{2}{C}\sum_{n=1}^{N}\sum_{c=1}^{C}(\alpha_c - 1)\log(u_{nc}) \\ &+ \frac{1}{C}\sum_{c=1}^{C}(\mathbf{y}_c - \boldsymbol{\mu}_y)^T \boldsymbol{\Sigma}_y^{-1}(\mathbf{y}_c - \boldsymbol{\mu}_y) + \text{Penalty}(C). \quad (20) \end{aligned}$$

Treating this basic form of this objective as the prototype for the negative log-likelihood, we derive a new probabilistic model, which we call the Infinite Bayesian Fuzzy Clustering (IBFC) model. This is named in the style of many Bayesian

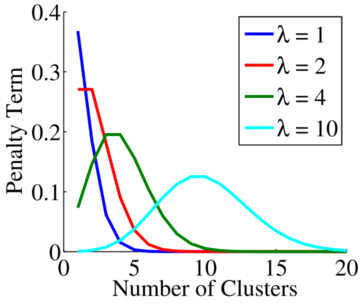


Fig. 8. Poisson prior on the number of clusters with $\lambda = 1, 2, 4, 10$ and the number of clusters ranging from 1 to 20.

nonparametric models such as infinite GMM (Dirichlet Process) [18] or the infinite latent feature model (Indian Buffet Process) [19], though these models are taken as the infinite limit over all model sizes, whereas the IBFC model simply averages over the unbounded model size.

The IBFC model consists of a data likelihood (21)

$$p(\mathbf{X}|\mathbf{U},\mathbf{Y},C) = \prod_{n=1}^{N} \frac{1}{Z_{\mathbf{X}}} \prod_{c=1}^{C} \mathcal{N}\left(\mathbf{x}_n | \boldsymbol{\mu} = \mathbf{y}_c, \boldsymbol{\Lambda} = \frac{1}{C} u_{nc}^{m} I\right) \tag{21}$$

a prior distribution over membership values (22),

$$\tilde{p}(\mathbf{U}|\mathbf{Y},C) = \prod_{n=1}^{N} \left[Z_{\mathbf{X}} \left(\prod_{n=1}^{C} \frac{u_{nc}^{m}}{C}^{\frac{D}{2}} \right) \text{Dirichlet}(\mathbf{u}_n|\boldsymbol{\alpha})^{\frac{1}{C}} \right] \times \left(C^{-C\frac{D}{2}} \right) \exp\left\{ \frac{\beta N}{C} \right\} \tag{22}$$

a Gaussian prior on the cluster prototypes (23) that is geometrically averaged over the model size,

$$p(\mathbf{Y}|C) = \prod_{c=1}^{C} \mathcal{N}(\mathbf{y}_c | \boldsymbol{\mu}_y, C\boldsymbol{\Sigma}_y) \tag{23}$$

and finally a Poisson prior on the number of clusters (24),

$$P(C) = \text{Poisson}\ (C|\lambda) = \frac{\lambda^C}{C!} \exp\{-\lambda\}. \tag{24}$$

The first step in creating this IBFC model is including the number of clusters C as a random variable, giving it a Poisson prior (24). The Poisson prior for the number of clusters was chosen for the following reasons: 1) It has a probabilistic form which fits well with the hierarchical Bayesian model employed; 2) the distribution is a discrete distribution defined over the positive integers (which is appropriate for the number of clusters); and 3) the range of the preferred number of clusters can be tuned using the λ parameter as illustrated in Fig. 8. The parameter of the Poisson can then be set in an empirical fashion to be increasing with the number of data points, for example in our applications, we choose $\lambda = \log(N)$.

The data likelihood of the IBFC model (21) is similar to previous BFC likelihood, but the averaging over the model size now appears in the precision parameter. The change in the precision value affects the normalization constant, and this is canceled again by the first two factors of the IBFC fuzzy cluster prior (IFCP) (22).

The arithmetic mean in the log-likelihood can be interpreted as a geometric mean in the regular likelihood space. We use this fact for the Dirichlet factor of the IFCP. The IFCP also has three additional multiplicative factors beyond the FCP from (5). The fourth multiplicative factor $F_4 = \left(C^{-CD/2}\right)$ is used to cancel the normalization terms of the cluster center prior (23) in a manner similar to that used for the data likelihood by F_2.

The fifth factor in the IFCP is used to promote model sparsity. It comes from placing an additional Laplace distribution on the memberships. The Laplace distribution, with mean parameter μ and scale parameter β, has the probability density, $p(x) = \beta/2 \exp\{-\beta|x-\mu|\}$. The Laplace prior with $\mu = 1$ rewards memberships being high and, thus, promotes fewer clusters.

Combining the first four factors of the IFCP with a Laplace distribution with $\mu = 1$ for each membership, and taking the geometric mean over the model size, results in

$$\tilde{p}(\mathbf{U}|Y,C) = \prod_{n=1}^{N} \left[Z_{\mathbf{X}} \left(\prod_{n=1}^{C} \frac{u_{nc}^{m}}{C}^{-\frac{D}{2}} \right) \text{Dirichlet}(\mathbf{u}_n|\boldsymbol{\alpha})^{\frac{1}{C}} \right] \times \left(C^{-C\frac{D}{2}} \right) \prod_{n=1}^{N} \prod_{c=1}^{C} \left(\beta/2 \exp\{-\beta|u_{nc}-1|\} \right)^{\frac{1}{C}}.$$

Focusing on the Laplace portion only, because the memberships will always be in the interval $[0, 1]$ and sum to 1 over all clusters, we can rewrite this as

$$\prod_{c=1}^{C} \left(\frac{\beta}{2} \exp\{-\beta(1-u_{nc})\} \right)^{\frac{1}{C}} = \frac{\beta}{2} \exp\left\{ -\frac{\beta}{C} \sum_{c=1}^{C} (1-u_{nc}) \right\} = \frac{\beta}{2} \exp\{-\beta\} \exp\left\{ \frac{\beta}{C} \right\}.$$

Considering now the product over all data points gives

$$\prod_{n=1}^{N} \frac{\beta}{2} \exp\{-\beta\} \exp\left\{ \frac{\beta}{C} \right\} = \left(\frac{\beta}{2} \right)^{N} \exp\{-\beta N\} \exp\left\{ \frac{\beta N}{C} \right\}.$$

Finally, removing the unneeded constants $(\beta/2)^N$ and $\exp\{-\beta N\}$ yields the last factor, $\exp\left\{\frac{\beta N}{C}\right\}$, of the IFCP prior (22).

The log-joint-likelihood of the IBFC model, formed from the log of the product of (21), (22), (23), and (24) can then be found to be

$$\begin{aligned} \log p(\mathbf{X},\mathbf{U},\mathbf{Y},C) \propto & -\frac{1}{2C} \sum_{n=1}^{N} \sum_{c=1}^{C} u_{nc}^{m} \|\mathbf{x}_n - \mathbf{y}_c\|^2 \\ & + \frac{1}{C} \sum_{n=1}^{N} \log \text{Dirichlet}(\mathbf{u}_n|\boldsymbol{\alpha}) + \frac{\beta N}{C} \\ & - \frac{1}{2C} \sum_{c=1}^{C} (\mathbf{y}_c - \boldsymbol{\mu}_y)^T \boldsymbol{\Sigma}_y^{-1} (\mathbf{y}_c - \boldsymbol{\mu}_y) \\ & + C \log\lambda - \sum_{i=1}^{C} \log(i). \end{aligned} \tag{25}$$

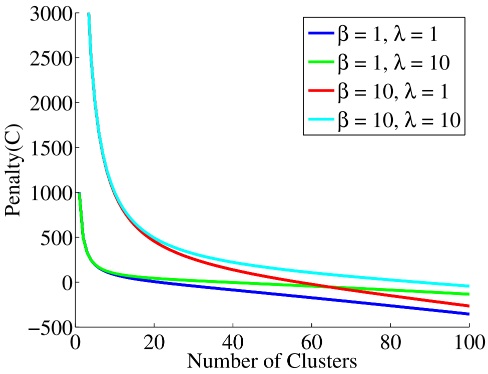


Fig. 9. Penalty term on the number of clusters with $\beta = 1, 10$, $\lambda = 1, 10$, $N = 1000$, and the number of clusters ranging from 1 to 100. Larger values indicate a higher likelihood and are preferred by the penalty term.

Taking the negative of this equation gives an objective function formulation which matches the proposed objective in (20), where the prior on C and the model sparsity terms combine to become the proposed penalty function.

Fig. 9 illustrates the effect of the proposed penalty function, $\frac{\beta N}{C} - \sum_{i=1}^{C} \log(i) + C \log \lambda - \lambda$. A small number of clusters are strongly encouraged. As the number of clusters increase, a larger penalty in incurred. The β parameter most strongly effects the penalty terms at a small number of clusters, whereas the penalty rate for a large number clusters is mostly determined by the λ parameter.

VI. Particle Filter Algorithm for Approximate Maximum-*a-Posteriori* Inference in the Infinite Bayesian Fuzzy Clustering Model

We present a particle filter inference algorithm called the IBFC Particle Filter (IBFC-PF) for finding local MAP parameter values in the IBFC model. One of the most well-known instances of a particle filter is in the Condensation algorithm [20] designed for object tracking, and as such, particle filters are most often associated with dynamic models having a time component. The particle filter can also be applied for approximate inference of the parameters of a static model [21], and it is used in that mode here.

A particle filter represents a probability density function using a discrete approximation. This approximation consists of a set of sample values on the input domain of the function, known as particles and the output value of the density function for each sample, known as the particle's weight. The filter part of the name comes from its similarity to the Kalman filter in tracking applications. A particle filter consists of an iteration of four basic steps:

1) *Deterministic Drift:* The particle state is updated according to system's time dynamics (e.g., equations of motion).
2) *Random Diffusion:* The particle state is updated according to system's random dynamics (e.g., random position/direction change).
3) *Weighting:* The particles are weighted according to their likelihood given the current system observation (e.g., measurement noise).
4) *Resampling:* The particle population is resampled (with replacement) by probabilities equal to the normalized particle weights.

To apply the filter process to MAP search in a static model, the drift step becomes a closed-form conditional maximization of a subset of the state variables. The diffusion step becomes a random sampling of the variables for which no closed-form maximization is available. The weighting step evaluates the likelihood of the particles, and then the resampling step repopulates the particles with a variety of good answers. The particle filter search can also be thought of as a parallel version of a stochastic expectation-maximization approach.

The IBFC-PF algorithm, shown in Algorithm 2, uses a mix of stochastic and closed-form updates. In order to use the closed-form updates, this algorithm requires that the Dirichlet concentration parameter be set to $\boldsymbol{\alpha} = 1$ and the fuzzifier be $m > 1$.

After initialization, the search loops until a maximum number of iterations or convergence have been reached. In each iteration, the particle filter goes through the four phases of random diffusion, deterministic drift, weighting, and then resampling. Performing the random step before the deterministic step here speeds convergence and simplifies the implementation.

In the first phase corresponding to the random diffusion, a random new model size C is drawn from a Poisson distribution with a mean value at the particle's current size, and then the cluster prototypes are changed accordingly. If the new size is greater than the current, new cluster prototypes are sampled from their prior. If the new size is smaller than the current, a random subset of the cluster prototypes are retained.

Next the memberships and cluster prototypes are updated, this is the deterministic drift step. The membership update uses the same closed-form update as traditional FCM (2). For the cluster prototypes, however, the addition of the prior distribution regularizes the objective function. Thus, a closed-form update for new prototypes must take this into account. Taking the log-likelihood of the IBFC model, then taking the partial derivative of the log-likelihood with respect to the prototype and setting it equal to zero yields the new update equation

$$\mathbf{y}_c = \left(\sum_{n=1}^{N} u_{nc}^{m}\mathbf{I} + \boldsymbol{\Sigma}_y^{-1}\right)^{-1}\left(\sum_{n=1}^{N} u_{nc}^{m}\mathbf{x}_n + \boldsymbol{\mu}_y\right). \qquad (26)$$

In the third phase, each particle is first weighted by the exponentiation of its log-joint-likelihood (25), as this is proportional to the log-posterior-likelihood of the parameters given the data. Before resampling, each particle is checked to see if it improves the MAP estimate for that model size. If so, it is retained as a candidate for the final MAP solution.

The particles are then sampled (with replacement) from the list of particles concatenated with the list of potential MAP solutions. As this is a search algorithm (and not a tracking filter), the addition of the MAP solutions in the resampling speeds convergence by encouraging the search to look near the current best answers.

Convergence of the particle filter is determined by the change in the likelihood value of the most likely MAP sample being below a threshold for multiple iterations in a row. Because of

Algorithm 2: Particle Filter IBFC MAP Search

```
Data: Data matrix $\mathbf{X}$, fuzzifier $m$, number of particles $P$, number of sampling iterations $N_{\text{iter}}$
Result: MAP estimates for number of clusters $C^*$, memberships $\mathbf{U}^*$ and cluster prototypes $\mathbf{Y}^*$
initialize hyperparameters $\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y$ from (9) and (10)
create initial sample $s$, where $s.C \leftarrow 1$, $s.u_{n1} \leftarrow 1$ for all $n = 1 \ldots N$, $s.\mathbf{y}_1 \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$
evaluate sample log-likelihood $s.\text{ll}$ from (25)
set initial MAP estimate for model size 1, $\text{MAP}[1] \leftarrow s$
initialize particles $\text{PCL}[i] \leftarrow s$ for all $i = 1 \ldots P$
for $\text{iter} = 1 \ldots N_{\text{iter}}$ do
    for $i = 1 \ldots P$ do
        sample new model size $C \sim \text{Poisson}(\text{PCL}[i].C)$
        if $C < \text{PCL}[i].C$ then
            $\text{inds} \leftarrow$ random subset length $C$ of range $1 \ldots \text{PCL}[i].C$
            $\text{PCL}[i].\mathbf{Y} \leftarrow \text{PCL}[i].\mathbf{Y}[\text{inds}]$
        else if $C > \text{PCL}[i].C$ then
            $N_{\text{new}} \leftarrow C - \text{PCL}[i].C$
            sample new prototypes $\mathbf{y}_k^{(\text{new})} \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ for $k = 1 \ldots N_{\text{new}}$
            $\text{PCL}[i].\mathbf{Y} \leftarrow [\text{PCL}[i].\mathbf{Y},\ \mathbf{Y}^{(\text{new})}]$
        $\text{PCL}[i].C \leftarrow C$
        update memberships $\text{PCL}[i].\mathbf{U} \leftarrow$ U-update$(X, \text{PCL}[i].\mathbf{Y})$ from (2)
        update cluster prototypes $\text{PCL}[i].\mathbf{Y} \leftarrow$ Y-update$(X, \text{PCL}[i].\mathbf{U})$ from (26)
        evaluate particle log-likelihood $\text{PCL}[i].\text{ll}$ from (25)
        if $\text{PCL}[i].\text{ll} > \text{MAP}[C].\text{ll}$ then
            $\text{MAP}[C] \leftarrow \text{PCL}[i]$
    construct list of particles for resampling $\text{PCLS} \leftarrow [\text{PCL},\ \text{MAP}]$
    for $i = 1 \ldots P$ do
        $\text{ind} \leftarrow \text{sample}(\exp\{[\text{PCLS.ll}]\} / \sum[\exp\{\text{PCLS.ll}\}])$
        $\text{PCL}[i] \leftarrow \text{PCLS}[\text{ind}]$
    stop if converged
pick MAP model size, $C \leftarrow \text{argmax}([\text{MAP.ll}])$
$C^* \leftarrow C,\ \mathbf{U}^* \leftarrow \text{MAP}[C].\mathbf{U},\ \mathbf{Y}^* \leftarrow \text{MAP}[C].\mathbf{Y}$
```

the random nature of the search, the samples may converge to a local maximum, but then not be improved upon for an indeterminate number of iterations. Eventually, through the random adding and removing of cluster prototypes, however, a new more likely area of parameter space is found, and improvements begin again. Thus, assessment of convergence of the algorithm is difficult. We have found that for cases with few cluster in low dimensions, a small number of iterations near 10 after convergence is sufficient, while for high-dimensional problems with many clusters, a larger number near 100 is often needed. Our synthetic data experiments in Section VII-A reached convergence in a few tens of iterations for small problems and fewer than 500 iterations for larger ones.

Because of the random selection of model size and new cluster initializations, the IBFC-PF algorithm as given is more likely to find a good maximum value than an algorithm which uses no randomness. However, it is not guaranteed to be able to explore the entire solution state space and, therefore, cannot guarantee a global optimum answer. If in the random portion of the particle update, the cluster prototypes were randomly reinitialized with some low probability, this would be enough to guarantee accessibility to all of the state space. Because the exploration of the space would be exceedingly slow, however, this step was deemed unnecessary. Thus, convergence is only guaranteed to a local optima.

We also note that, instead of the hybrid deterministic/stochastic particle filter, a full MCMC sampling approach could be derived for the IBFC model as well. Such an algorithm would allow the Dirichlet concentration to be varied and the fuzzifier set to exotic values, as well as guarantee convergence to a global optimum. However such an algorithm would be much slower than the IBFC-PF in operation.

The asymptotic computational complexity of a single iteration of this algorithm is $O(PNCD + PCD^3)$, where P is the number of particles, N is the number of data points, C is the number of clusters, and D is the dimensionality of the data. The D^3 comes from the matrix inverse in (26). Depending upon the inversion implementation used, this exponent could be lower, although still greater than 2. If the implementation is restricted to use a diagonal covariance in the cluster prototype prior, then each iteration will scale at $O(PNCD)$.

A. Parameter Settings

The IBFC-PF algorithm has five parameters that must be set prior to running, discussed as follows.

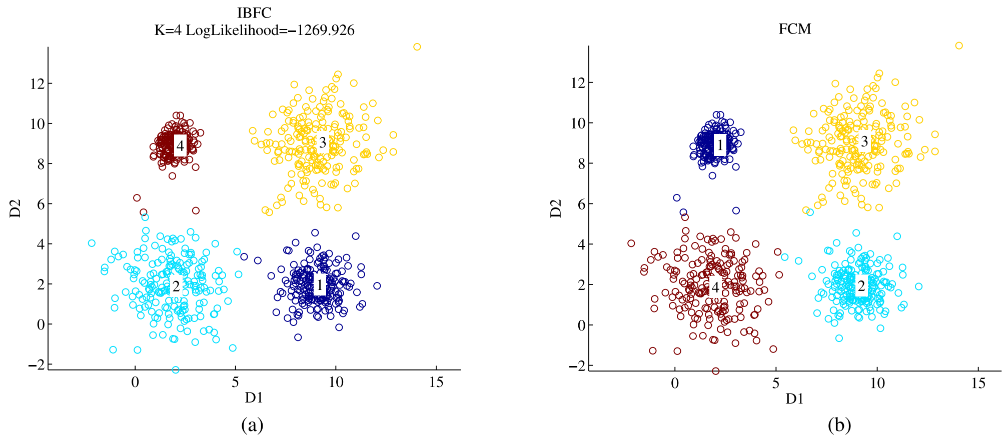


Fig. 10. IBFC and FCM results on four-component data, color coded by maximum membership cluster. (a) IBFC-PF $m = 2, \boldsymbol{\alpha} = 1, \beta = 2$. (b) FCM $m = 2, C = 4$.

1) *Model Sparsity Weight—β:* The β parameter is a key parameter in the IBFC model and plays a significant role in determining the number of clusters. As β increases, a smaller number of clusters tends to be found. The stability of the output of the IBFC algorithm in terms of the number of clusters is dataset dependent. Datasets with distinct, well separated clusters are less sensitive to this parameter. Fig. 9 illustrates how changes to the β parameter affect the penalty term on the number of clusters.
2) *Number of Clusters Prior Parameter—λ:* The λ parameter controls the shape of the Poisson prior on the number of clusters. The number of clusters preferred by the prior distribution is equal to λ. In practice, we set $\lambda = \log(N)$. Fig. 9 illustrates how changes to the λ parameter effect the penalty term on the number of clusters.
3) *Number of Particles:* The number of particles affect the per-iteration time and convergence rate. A larger number of particles results in a more thorough exploration of the search space but a longer running time per iteration.
4) *Fuzzifier—m:* The m parameter plays the same role as the fuzzifier in the standard FCM algorithm. Use of this parameter is the same as the BFC MAP search and is discussed in Section III-A.
5) *Prototype Prior Variance Scale—γ:* Use of this parameter is the same as the BFC MAP search and is discussed in Section III-A.

VII. Experiments and Comparisons With Other Algorithms

To demonstrate the ability of the IBFC model and particle filter inference, we constructed a 2-D synthetic dataset of 200 random samples from each of four Gaussian components with the following means and covariances: $\boldsymbol{\mu}_1 = (2\ \ 2)^T, \boldsymbol{\mu}_2 = (2\ \ 9)^T, \boldsymbol{\mu}_3 = (9\ \ 2)^T, \boldsymbol{\mu}_4 = (9\ \ 9)^T$, and $\boldsymbol{\Sigma}_1 = 2\,\mathbf{I}_{2\times 2}$, $\boldsymbol{\Sigma}_2 = 0.25\,\mathbf{I}_{2\times 2}, \boldsymbol{\Sigma}_3 = 0.75\,\mathbf{I}_{2\times 2}, \boldsymbol{\Sigma}_4 = 2\,\mathbf{I}_{2\times 2}$.

Fig. 10(a) shows the results obtained by running the IBFC-PF on the four-component dataset. The IBFC-PF found four components to be the most likely solution with a log-likelihood of −1269. Fig. 10(b) shows the FCM results for this data with the correct number of clusters specified. Alternative solutions with other models sizes were evaluated by the IBFC-PF, with maximal log-likelihoods as follows: $C = 2 \rightarrow -2965$, $C = 3 \rightarrow -1703$, $C = 5 \rightarrow -1289$, and $C = 6 \rightarrow -1302$.

For application to real-world data, the IBFC-PF algorithm was applied to a high-dimensional hyperspectral image dataset. The dataset used is the MUUFL Gulfport hyperspectral data, collected over the University of Southern Mississippi-Gulf Park campus [22]. Fig. 11(a) shows an RGB image of this scene. Each pixel corresponds to $1\,m^2$. Data were collected with an ITRES Inc. hyperspectral Compact Airborne Spectrographic Imager (CASI-1500) over the 375–1050 nm range with 72 10-nm spectral bands. The image is 284×271 pixels in size. Thus, it contains 76 964 data points of dimensionality 72.

When applying IBFC-PF to this data with $\beta = 0.1$ and $m = 2$, it estimated five clusters. Fig. 11(b) shows the cluster label for each pixel across the scene with the highest membership value. As can be seen by comparing the label image to the RGB, the five clusters roughly correspond to the following regions: 1) asphalt, 2) dirt/soil, 3) concrete/bright sand, 4) dark trees and shadow, and 5) grass.

Next, the hyperspectral data was appended with the row and column associated with each pixel (resulting in 74-D data). The IBFC-PF was then applied to this modified dataset with the same parameters as before, $\beta = 0.1$ and $m = 2$. The result from this experiment is shown in Fig. 11(c). This time the algorithm found 35 clusters. As can be seen in Fig. 11(c), many more clusters are used than in Fig. 11(b), and the clusters now contain pixels that are nearby spatially as well as spectrally.

A. Comparison to Competitive Agglomeration

To evaluate the effectiveness of the IBFC-PF algorithm in identification of the number of clusters in the data, we ran a synthetic data experiment with randomly generated clusters. We, then, ran the same experiments with the CA fuzzy clustering algorithm, which also estimates the number of clusters, in order to provide a comparison.

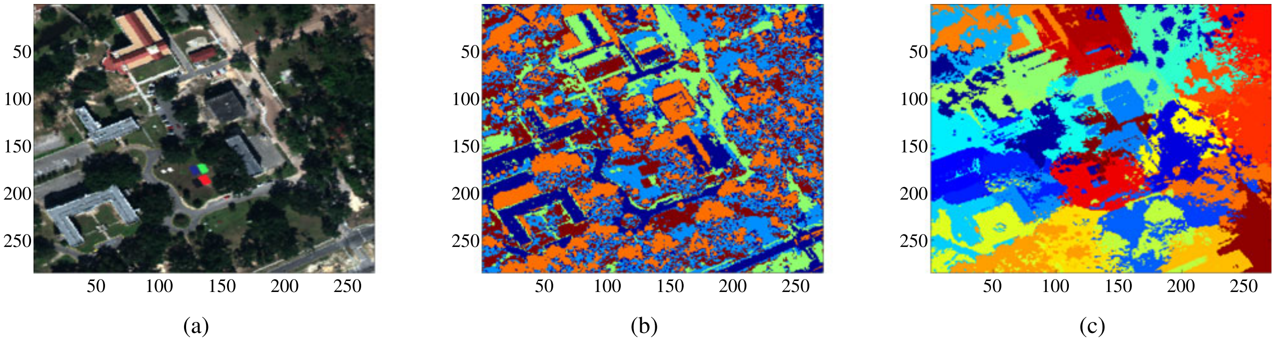


Fig. 11. (a) RGB image of the MUUFL Gulfport hyperspectral dataset collected over the University of Southern Mississippi Gulf-park campus. IBFC-PF results of the MUUFL Gulfport hyperspectral dataset using $\beta = 0.1$ and $m = 2$, for (b) clustering spectral dataset only, and (c) clustering spatial-spectral dataset with appended row/column information.

TABLE I
Average Rand Index for Competitive Agglomeration and IBFC Versus Truth (Larger is Better)

$D \downarrow$	$C \rightarrow$	2	3	4	5	6	7	8	9	10
2	CA	0.927	0.931	0.931	0.932	0.944	0.927	0.933	0.909	0.918
	IBFC	0.992	0.960	0.951	0.951	0.962	0.944	0.946	0.947	0.948
3	CA	0.979	0.982	0.987	0.979	0.977	0.963	0.947	0.944	0.923
	IBFC	0.998	0.991	0.993	0.992	0.986	0.974	0.978	0.981	0.978
4	CA	0.990	0.996	0.993	0.992	0.987	0.961	0.977	0.962	0.901
	IBFC	1.000	0.999	0.999	0.997	0.991	0.991	0.988	0.989	0.984
5	CA	0.993	0.999	0.984	0.969	0.934	0.988	0.962	0.948	0.932
	IBFC	1.000	0.999	0.997	1.000	0.993	0.996	0.996	0.996	0.993
6	CA	0.982	0.975	0.949	0.893	0.969	0.965	0.955	0.972	0.967
	IBFC	1.000	1.000	1.000	1.000	0.997	0.995	0.998	0.995	0.995
7	CA	0.952	0.954	0.950	0.999	0.860	0.939	0.966	0.905	0.978
	IBFC	1.000	1.000	1.000	1.000	1.000	1.000	0.998	0.999	0.997
8	CA	0.894	0.951	0.850	0.892	0.916	0.971	0.937	0.991	0.917
	IBFC	1.000	1.000	1.000	1.000	1.000	1.000	0.999	0.998	0.999
9	CA	0.847	0.950	0.802	0.787	0.916	0.999	0.910	0.970	0.992
	IBFC	1.000	1.000	1.000	1.000	1.000	1.000	0.999	0.999	0.998
10	CA	0.800	0.822	0.751	0.893	0.886	0.999	0.884	0.908	0.879
	IBFC	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.999	1.000

TABLE II
Average EMD for Competitive Agglomeration and IBFC Versus Truth (Smaller is Better)

$D \downarrow$	$C \rightarrow$	2	3	4	5	6	7	8	9	10
2	CA	0.59	0.79	0.93	1.11	1.22	1.47	1.52	1.78	2.01
	IBFC	0.35	0.60	0.81	0.97	1.05	1.28	1.38	1.48	1.66
3	CA	0.52	0.79	0.95	1.21	1.42	1.63	2.04	2.40	3.04
	IBFC	0.44	0.71	0.89	1.07	1.33	1.51	1.67	1.87	2.09
4	CA	0.55	0.80	1.02	1.25	1.52	2.06	2.16	2.62	3.72
	IBFC	0.51	0.76	0.99	1.21	1.48	1.69	1.94	2.12	2.46
5	CA	0.57	**0.84**	1.15	1.65	2.26	1.98	2.46	3.09	3.73
	IBFC	0.55	0.85	1.09	1.32	1.62	1.82	2.07	2.34	2.59
6	CA	0.88	1.30	1.74	2.85	2.05	2.44	2.99	3.14	3.67
	IBFC	0.62	0.92	1.18	1.45	1.74	1.99	2.23	2.53	2.77
7	CA	1.61	1.60	2.07	**1.52**	3.60	2.87	2.89	3.86	3.57
	IBFC	0.66	0.99	1.27	1.53	1.83	2.05	2.43	2.61	2.94
8	CA	2.68	1.87	3.94	3.40	3.17	2.58	3.51	2.99	4.77
	IBFC	0.69	1.03	1.33	1.67	1.90	2.20	2.52	2.76	3.02
9	CA	3.83	1.77	5.01	5.26	3.33	**2.30**	4.05	3.41	3.56
	IBFC	0.77	1.08	1.42	1.74	2.09	2.32	2.68	2.96	3.23
10	CA	5.49	4.57	6.20	3.68	4.13	**2.47**	4.54	4.64	6.09
	IBFC	0.75	1.13	1.46	1.82	2.15	2.48	2.79	3.16	3.39

In the experiment, we varied the number of clusters from 2 to 10 and the dimensionality of the data from 2 to 10. For each cluster, a random mean was drawn from the hypercube on the interval $[-10, 10]$ for each dimension. Then, a random number of samples was selected uniformly from 30 to 100, and that number of samples was drawn from a Gaussian with the identity as covariance. For these datasets, especially in lower dimensional data, some of the clusters will overlap, and the algorithms are not expected to always find the same number of clusters as was used to generate the dataset.

For each number of clusters and dimensionality, 30 trials were conducted. In each trial, a new dataset was generated, then both IBFC-PF and CA were run on the dataset. The algorithms were then scored using the Rand cluster validity index [23], the Earth Mover's Distance (EMD) cluster validity metric proposed by Anderson *et al.* [24], and the fraction of trials in which the algorithm returned the same number of clusters as was used to generate the dataset.

The IBFC-PF used the parameter settings $m = 2$ for the fuzzifier, sparsity weight equal to the dimensionality of the data $\beta = D$, number of particles $P = 10$, maximum number of iterations $N_{\text{iter}} = 500$, and (implicitly for the IBFC-PF) the Dirichlet concentration set to $\boldsymbol{\alpha} = 1$. The stopping condition was no change greater than 10^{-5} for ten iterations with dimensionality 2 through 7, or for 100 iterations with dimensionality 8 or higher.

The CA algorithm used the settings, fuzzifier $m = 2$, the initial average distance to cluster cardinality ratio weighting of $\eta_0 = 5$, an exponential weighting decay time constant $\tau = 10$, maximum of 50 iterations, initial number of clusters $C_{\text{init}} = 20$. These parameter settings are likely suboptimal at some points in the range of tested datasets, but they were chosen such that they had subjectively good performance in initial testing, and because they are based on the methods discussed in the original paper [3] likely to be the first settings attempted in any use of the algorithm.

Table I shows the average Rand index between both the CA output and the true generating components, and between the IBFC-PF and the true number of components. The number of dimensions D of the dataset increases along the rows of the

TABLE III
FRACTION OF TRIALS WITH SAME NUMBER OF CLUSTERS AS TRUTH (LARGER IS BETTER) AND AVERAGE NUMBER OF CLUSTERS FOUND (IN PARENTHESES)

$D \downarrow$	$C \rightarrow$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
2	**CA**	0.43 (2.9)	0.33 (3.9)	0.47 (4.4)	0.60 (5.1)	0.40 (5.5)	0.10 (5.9)	0.13 (6.9)	0.03 (6.3)	0.00 (7.3)
	IBFC	1.00 (2.0)	0.90 (2.9)	0.80 (3.8)	0.70 (4.7)	0.63 (5.6)	0.63 (6.0)	0.27 (6.8)	0.07 (7.4)	0.10 (7.9)
3	**CA**	0.70 (2.3)	0.63 (3.3)	0.83 (4.1)	0.67 (5.2)	0.57 (5.7)	0.33 (6.5)	0.20 (6.8)	0.13 (7.3)	0.00 (7.0)
	IBFC	1.00 (2.0)	0.97 (3.0)	0.97 (4.0)	0.93 (4.9)	0.80 (5.8)	0.80 (6.5)	0.43 (7.4)	0.40 (8.2)	0.20 (8.8)
4	**CA**	0.83 (2.2)	0.80 (3.2)	0.90 (4.0)	0.90 (5.0)	0.77 (5.9)	0.73 (6.8)	0.47 (7.3)	0.30 (7.8)	0.03 (7.6)
	IBFC	1.00 (2.0)	1.00 (3.0)	1.00 (4.0)	0.97 (5.0)	0.83 (5.8)	0.83 (6.8)	0.63 (7.6)	0.53 (8.5)	0.33 (9.1)
5	**CA**	0.87 (2.1)	0.97 (3.0)	0.73 (4.1)	0.93 (5.0)	0.73 (6.3)	0.73 (6.8)	0.67 (7.9)	0.47 (8.3)	0.37 (8.6)
	IBFC	1.00 (2.0)	1.00 (3.0)	0.97 (4.0)	1.00 (5.0)	0.87 (5.9)	0.87 (6.9)	0.80 (7.8)	0.77 (8.8)	0.57 (9.5)
6	**CA**	0.93 (2.3)	0.83 (3.5)	0.83 (4.6)	0.83 (5.7)	0.87 (6.1)	0.73 (7.2)	0.70 (7.8)	0.60 (8.2)	0.43 (8.8)
	IBFC	1.00 (2.0)	1.00 (3.0)	1.00 (4.0)	1.00 (5.0)	0.93 (5.9)	0.93 (6.9)	0.90 (7.9)	0.73 (8.7)	0.73 (9.7)
7	**CA**	0.90 (2.5)	0.90 (3.4)	0.90 (4.8)	0.97 (5.0)	0.73 (6.8)	0.83 (7.3)	0.73 (8.1)	0.57 (8.7)	0.60 (9.3)
	IBFC	1.00 (2.0)	1.00 (3.0)	1.00 (4.0)	1.00 (5.0)	1.00 (6.0)	1.00 (7.0)	0.93 (7.9)	0.93 (8.9)	0.80 (9.8)
8	**CA**	0.63 (3.0)	0.77 (3.5)	0.67 (5.0)	0.77 (5.6)	0.70 (6.5)	0.90 (7.1)	0.83 (8.2)	0.83 (8.8)	0.50 (8.9)
	IBFC	1.00 (2.0)	1.00 (3.0)	1.00 (4.0)	1.00 (5.0)	1.00 (6.0)	1.00 (7.0)	0.97 (8.0)	0.90 (8.9)	0.93 (9.9)
9	**CA**	0.60 (3.7)	0.73 (3.5)	0.63 (5.6)	0.60 (6.2)	0.70 (6.5)	0.97 (7.0)	0.77 (8.4)	0.93 (9.0)	0.67 (9.6)
	IBFC	1.00 (2.0)	1.00 (3.0)	1.00 (4.0)	1.00 (5.0)	1.00 (6.0)	1.00 (7.0)	0.97 (8.0)	0.97 (9.0)	0.93 (9.9)
10	**CA**	0.57 (4.3)	0.63 (4.4)	0.63 (5.8)	0.83 (5.7)	0.73 (6.7)	0.87 (7.1)	0.77 (8.5)	0.80 (8.9)	0.47 (9.2)
	IBFC	1.00 (2.0)	1.00 (3.0)	1.00 (4.0)	1.00 (5.0)	1.00 (6.0)	1.00 (7.0)	1.00 (8.0)	0.93 (8.9)	1.00 (10.0)

table, and the number of components C in the generating dataset increases along the columns. To compute this index, first the fuzzy memberships computed by IBFC-PF and CA were converted to crisp cluster assignments by maximum membership. The results here show that, on average, for every tested dimensionality and number of clusters, the IBFC-PF algorithm produced a clustering that was as similar to or more similar to the true generating components than did CA.

The Rand index is just one of many methods of measuring clustering similarity, and it does not take into account fuzzy memberships or the location of cluster prototypes. To take the memberships and prototypes into account, we used a version of Anderson *et al.*'s EMD measure of fuzzy clustering similarity [24]. We chose to use the Euclidean distance between a cluster's prototype and the generating component's mean as the ground distance. This was done in order to explicitly take the accuracy of the cluster prototypes into account. Smaller numbers of this measure indicate less dissimilarity between the clustering output and the true generating components.

Table II shows the average EMD measure between the clustering output and generating components over the 30 trials at each D and C size. The results in boldface show where CA performed better in this measure on average than the IBFC-PF. In total, the IBFC-PF performed as well or better than CA in 77 of 81 tests (95%).

Table III shows the fraction of tests in which the clustering algorithms found the same number of clusters and the generating data, as well as the average number of clusters found by the algorithm. The IBFC-PF performed as well or better than CA all of the 81 tests. Both algorithms often did not correctly identify the number of clusters in low-dimensional data with a large number of clusters, due to the generated data overlapping and being indistinguishable as separate clusters. Looking at the results from any column, the IBFC-PF shows an encouraging trend of generally getting the number of clusters correct more often with increasing dimensionality. While CA on the other hand gets better with dimensionality to a point, but then performance generally falls off at the highest numbers of dimensions.

B. Comparison With Fuzzy C-Means With a Validity Metric

Furthermore, we assert that one advantage of the IBFC approach is the removal of the manual process of selecting the number of clusters that generally accompanies cluster validity methods. The use of internal validation metrics paired with a clustering algorithm generally requires visual comparison of the validity metrics in order to determine the number of clusters. However, even with manual visual interpretation, often, the number of clusters is extremely difficult to discern. The Xie-Beni (XB) index [25] is a common cluster validity metric for fuzzy clustering results [23]. In the literature, selection of the number of clusters with XB involves the difficult manual identification the "knee" of the XB curve computed across clustering results with a variable number of clusters.

In order to illustrate this, a series of experiments on synthetic data were conducted. The synthetic data was generated by, first, randomly generating a number of clusters. A random dimensionality was selected from 2 to 10, and then, for each cluster, a random mean was drawn from the hypercube on the interval $[-10, 10]$ for each dimension. A random number of samples was selected uniformly from 30 to 100, and that number of samples was drawn from a Gaussian with the identity as covariance. Given this data, the IBFC-PF algorithm was applied with β set to the number of dimensions and $m = 2$. In addition, FCM with $m = 2$ was applied to the data for 2 to 10 number of clusters. For each of the FCM results, the XB index was computed. Given the XB curves across the number of clusters, the "knee" of the XB curve was autonomously found by identifying the number of clusters in which the angle of the XB curve at that point was the smallest. In Fig. 12(a), the true clustering (projected on the first two dimensions), (b) the IBFC clustering result, (c) the FCM clustering result with $m = 2$ in which the number of clusters is estimated by identifying the "knee" XB curve, and (d) the

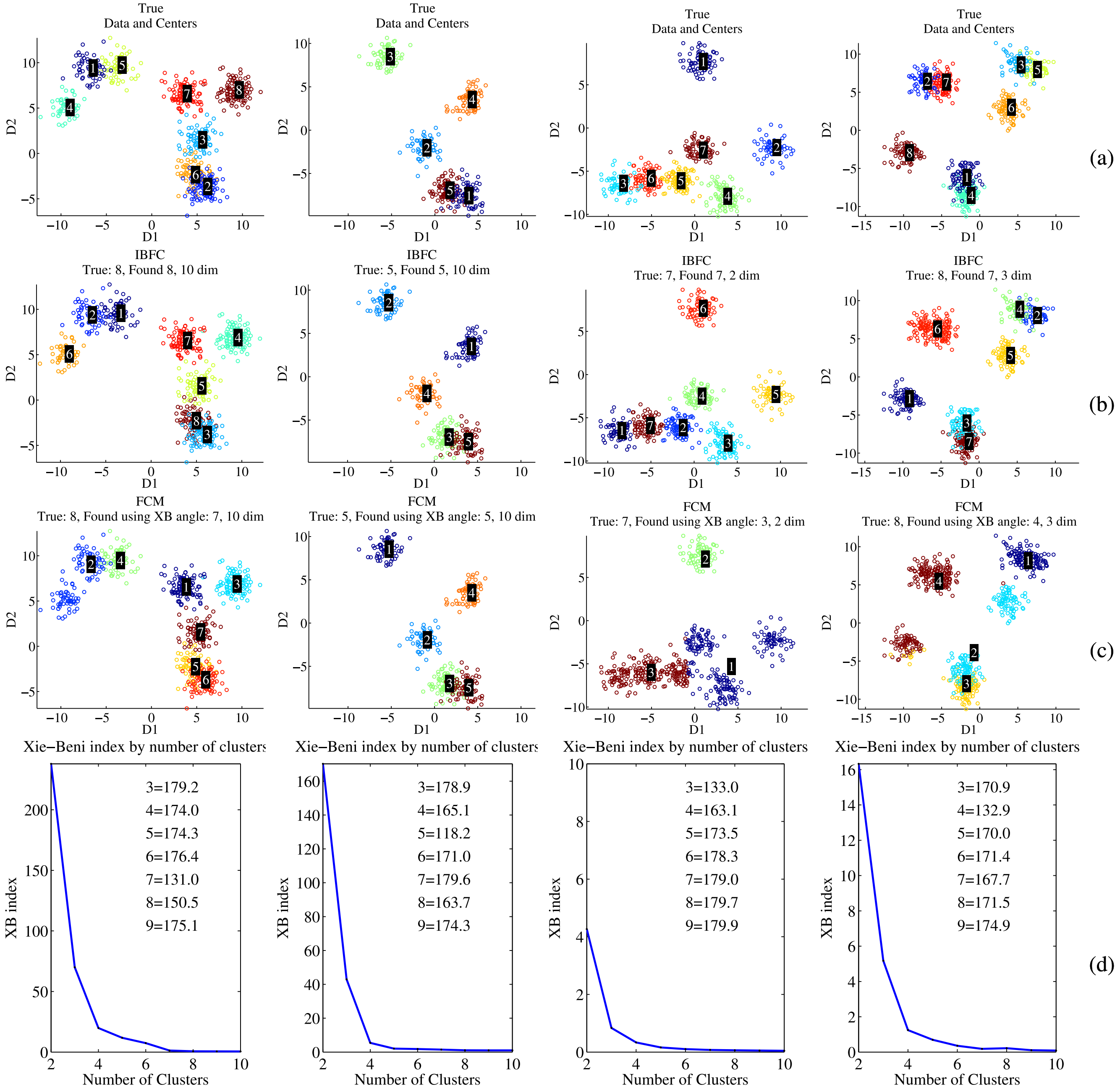


Fig. 12. Series of four experiments with randomly generated synthetic data comparing IBFC with the FCM and XB index. For each column of the results, (a) is the true clustering (projected on the first two dimensions), (b) is the IBFC clustering result, (c) is the FCM clustering result with $m = 2$ in which the number of clusters is estimated by identifying the "knee" XB curve, and (d) is the XB curve with the associated angles used to find the "knee" of the curve. Note that the axis scaling distorts the apparent angles of the plots.

XB curve with the associated angles used to find the "knee" of the curve are shown. As can be seen in the figure, the XB approach is far less capable of estimating the number of clusters as compared with the IBFC. Even with manual interpretation, the number of clusters is difficult to determine from the XB curves.

VIII. Discussion and Future Work

In this paper we have, first, derived a probabilistic model for fuzzy clustering. Developing this model has allowed us to use probabilistic inference techniques to learn the parameters of the model. Care was taken in designing the model such that the logarithm of the likelihood function (with the Dirichlet parameter set to $\alpha = 1$) becomes essentially the classical FCM objective. The probabilistic inference methods, thus, serve the role of global optimization of the nonlinear objective function. The same behaviors could equally well be achieved by constructing an appropriate objective function and using nonlinear optimization techniques. Some constraints such as positive definiteness for covariance matrices, or the mix of continuous variables and the integer number of clusters in the IBFC model, add challenging aspects to such optimization which are easily accommodated by probabilistic techniques.

Future work will include investigation into alternative forms for the prior on the number of clusters in the IBFC model. In particular, the Negative Binomial distribution will be investigated as it is defined over nonnegative integers and can be parameterized such that it has both a mean and variance. In contrast, the mean and variance of the Poisson distribution, which is used in the current model, are fixed to be equal.

In designing the IBFC model, we find that we have traded one parameter, the specified number of clusters, for two less interpretable ones, the model sparsity weight, and the Poisson prior hyperparameter. We must be careful that these parameters are indeed more useful than simply specifying the number of clusters. Our synthetic data experiments show that a single setting for the sparsity weight (the dimensionality of the data) and setting the Poisson parameter to $\lambda = \log(N)$, was sufficient for accurately learning the number of clusters over a moderate range. Future work will need to focus on the setting, sensitivity, and interpretation of these parameters, or indeed even exploration of other model sparsity promoting terms.

In addition, various methods can be implemented to improve the performance of IBFC-PF. Individual particles can be marked when they converge so that iterations that do not change the model size can be bypassed. If a particle is represented multiple times in the particle set, and its size is unchanged after the random sampling step, then those copies will duplicate the computation of the membership update. This extra work can be eliminated. Memory usage can be reduced by limiting the list of MAP estimates to retain only the top K most likely sizes. Additionally, if any particle is also present in the list of MAP samples, then it is overrepresented when resampling, which may slow the exploration of state space. This could be avoided with additional checks. Finally, the particles can trivially be updated in parallel.

An aspect which may improve convergence of the particle filter is removal of clusters in proportion to their cardinality. When a lower number of clusters is selected for a particle, the clusters could be randomly selected for removal in proportion to their cardinality. However, this might also restrict exploration of the solution space due to the retaining of large but poor clusters. Thus, analysis of such techniques is left for future work.

This paper has also only addressed compact spherical clusters, that is those modeled by a Gaussian with unit covariance. The extension of the model to elliptical clusters is straightforward by adding the covariance parameters and a Wishart prior distribution. The prior distribution on the covariances will act as regularization, allowing closed-form updates of the covariances in a IBFC-PF like algorithm without the use of correction factors as in the commonly seen modified Gustafson–Kessel approach [26]. Other cluster shapes could be achieved with a univariate zero mean Gaussian on the squared distance for that cluster shape. Future work should explore these approaches.

One aspect of probabilistic models is their use in random data generation, i.e., the generative model. For the BFC model, the FDL gives that the distribution of random data, given the set prototypes $\mathbf{Y}$ and a vector of fuzzy-set memberships $\mathbf{u}$, will be a Gaussian distribution. This suggests that random dataset could be generated by the model $\mathbf{y}_c \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \mathbf{u} \sim \mathrm{Dirichlet}(\boldsymbol{\alpha})$, $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu} = f(\mathbf{u}, \mathbf{Y}), \boldsymbol{\Sigma} = g(\mathbf{u}, \mathbf{Y}))$. It is interesting to think that running fuzzy clustering on a dataset generated in such a fashion may correctly find the cluster prototypes, but seems unlikely to find the correct membership values. This is because the generating model has added variance $g(\mathbf{u}, \mathbf{Y})$ around the mean location $f(\mathbf{u}, \mathbf{Y})$, but each data point estimates its membership vector independently which cannot account for the variance. Therefore, if a large group of samples were assumed to all share the same membership vector, then that membership vector could be estimated from the data. However, the estimate from a single data point will almost surely be incorrect.

An extension of BFC may be in estimating appropriate values of the fuzzifier parameter m. Traditionally, the likelihood/objective function will improve rapidly as the value of the fuzzifier is increased, making it difficult to compare the validity of two models with different fuzzifiers. A prior distribution for m could conceivably be constructed such that this tradeoff is balanced. Another alternative may be to drop the fuzzifier and instead control this distribution of memberships through the Dirichlet concentration parameters.

IX. Appendix

The normalization $Z(\mathbf{u}_n, m, \mathbf{Y})$ for the FDL (4) is derived by first expontiating the FCM objective (1) terms for a given data point $\mathbf{x}_n$ multiplied by a factor of $-\frac{1}{2}$. Assuming Euclidean distance yields

$$\exp\left\{-\frac{1}{2}\sum_{c=1}^{C} u_{nc}^m (\mathbf{x}_n - \mathbf{y}_c)^T (\mathbf{x}_n - \mathbf{y}_c)\right\}.$$

Assuming that this exponential expression is part of a Gaussian PDF $f(\mathbf{x}_n)$, complete the square within the exponential to find the parameters of the corresponding normal

$$\begin{aligned} f(\mathbf{x}_n) \propto\ & \exp\left\{-\frac{1}{2}\sum_{c=1}^{C} u_{nc}^m \left[\mathbf{x}_n^T\mathbf{x}_n - 2\mathbf{x}_n^T\mathbf{y}_c + \mathbf{y}_c^T\mathbf{y}_c\right]\right\} \\ \propto\ & \exp\left\{-\frac{1}{2}\left[\left(\sum_{c=1}^{C} u_{nc}^m\right)\mathbf{x}_n^T\mathbf{x}_n - 2\mathbf{x}_n^T\left(\sum_{c=1}^{C} u_{nc}^m\mathbf{y}_c\right)\right.\right. \\ & \left.\left. + \left(\sum_{c=1}^{C} u_{nc}^m\mathbf{y}_c^T\mathbf{y}_c\right)\right]\right\} \end{aligned}$$

setting $\boldsymbol{\mu} = \frac{1}{\sum_{c=1}^{C} u_{nc}^m}\left(\sum_{c=1}^{C} u_{nc}^m\mathbf{y}_c\right)$ and $\boldsymbol{\Lambda} = \sum_{c=1}^{C} u_{nc}^m\mathbf{I}$

$$\begin{aligned} f(\mathbf{x}_n) \propto \exp&\left\{-\frac{1}{2}\left[\mathbf{x}_n^T\boldsymbol{\Lambda}\mathbf{x}_n - 2\mathbf{x}_n^T\boldsymbol{\Lambda}\boldsymbol{\mu} + \boldsymbol{\mu}^T\boldsymbol{\Lambda}\boldsymbol{\mu}\right]\right\} \\ \times \exp&\left\{-\frac{1}{2}\left[\left(\sum_{c=1}^{C} u_{nc}^m\mathbf{y}_c^T\mathbf{y}_c\right) - \boldsymbol{\mu}^T\boldsymbol{\Lambda}\boldsymbol{\mu}\right]\right\} \end{aligned}$$

$$f(\mathbf{x}_n) = \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}, \boldsymbol{\Lambda}).$$

Next, the normalization $Z(\mathbf{u}_n, m, \mathbf{Y})$ factor can be solved for by equating

$$\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}, \boldsymbol{\Lambda}) = \frac{1}{Z(\mathbf{u}_n, m, \mathbf{Y})} \prod_{c=1}^{C} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu} = \mathbf{y}_c, \boldsymbol{\Lambda} = u_{nc}^{m}\,\mathbf{I})$$

$$Z(\mathbf{u}_n, m, \mathbf{Y}) = \frac{\prod_{c=1}^{C} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu} = \mathbf{y}_c, \boldsymbol{\Lambda} = u_{nc}^{m}\,\mathbf{I})}{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}, \boldsymbol{\Lambda})}$$

$$= \frac{\prod_{c=1}^{C} 2\pi^{-D/2} u_{nc}^{mD/2} \exp\left\{-\frac{1}{2} u_{nc}^{m} (\mathbf{x}_n - \mathbf{y}_c)^T (\mathbf{x}_n - \mathbf{y}_c)\right\}}{2\pi^{-D/2}\left(\sum_{c=1}^{C} u_{nc}^{m}\right)^{D/2} \exp\left\{-\frac{1}{2}(\mathbf{x}_n^T - \boldsymbol{\mu})^T \boldsymbol{\Lambda} (\mathbf{x}_n^T - \boldsymbol{\mu})\right\}}$$

which, using the expansion for the FCM objective in (27), can be simplified to

$$Z(\mathbf{u}_n, m, \mathbf{Y}) = (2\pi)^{-\frac{D}{2}(C-1)} \left(\prod_{c=1}^{C} u_{nc}^{m}\right)^{\frac{D}{2}} \left(\sum_{c=1}^{C} u_{nc}^{m}\right)^{-\frac{D}{2}}$$

$$\times \exp\left\{-\frac{1}{2}\left(\sum_{c=1}^{C} u_{nc}^{m} \mathbf{y}_c^T \mathbf{y}_c - \frac{||\sum_{c=1}^{C} u_{nc}^{m} \mathbf{y}_c||^2}{\sum_{c=1}^{C} u_{nc}^{m}}\right)\right\}.$$

References

[1] J. C. Bezdek, *Pattern Recognition With Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer, 1981.

[2] I. Gath and A. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 773–780, Jul. 1989.

[3] H. Frigui and R. Krishnapuram, "Clustering by competitive agglomeration," *Pattern Recog.*, vol. 30, no. 7, pp. 1109–1119, 1997.

[4] M. Li *et al.*,"Agglomerative fuzzy k-means clustering algorithm with selection of number of clusters," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 11, pp. 1519–1534, Nov. 2008.

[5] F. J. Theis, "Bayesian fuzzy clustering of colored graphs," in *Proc. Int. Conf. Latent Variable Anal. Signal Separation*, 2012, pp. 528–535.

[6] S.-W. Lee, Y. S. Kim, and Z. Bien, "A probabilistic cluster validity index for agglomerative Bayesian fuzzy clustering," in *Proc. Int. Conf. Comput. Intell. Model. Control Autom.*, 2008, pp. 368–373.

[7] S. W. Lee *et al.*,"Iterative Bayesian fuzzy clustering toward flexible icon-based assistive software for the disabled," *Inf. Sci.*, vol. 180, no. 3, pp. 325–340, Feb. 2010.

[8] K.-J. Kim, S.-H. Yoo, and S.-B. Cho, "Bayesian validation of fuzzy clustering for analysis of yeast cell cycle data," in *Proc. Knowl.-Based Intell. Inform. Eng. Syst.*, 2005, pp. 777–784.

[9] Y. Tang *et al.*, "Fuzzy naive bayes classifier based on fuzzy clustering," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, vol. 5, Oct. 2002.

[10] C. Borgelt, H. Timm, and R. Kruse, "Using fuzzy clustering to improve naive Bayes classifiers and probabilistic networks," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2000, pp. 53–58.

[11] H. Feng and D. E. A. Giles, "Bayesian fuzzy regression analysis and model selection: Theory and evidence," Department of Economics, University of Victoria, Econometrics Working Papers 0710, 2007.

[12] P. Marttinen *et al.*,"Bayesian clustering of fuzzy feature vectors using a quasi-likelihood approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 1, pp. 74–85, Jan. 2009.

[13] S. Miyamoto, H. Ichihashi, and K. Honda, *Algorithms for Fuzzy Clustering: Methods in C-Means Clustering With Applications*, vol. 229. New York, NY, USA: Springer, 2008.

[14] H. Ichihashi, K. Miyagishi, and K. Honda, "Fuzzy c-means clustering with regularization by K-L information," in *Proc. IEEE Conf. Fuzzy Syst.*, Dec. 2001, vol. 2, pp. 924–927

[15] K. Honda and H. Ichihashi, "Regularized linear fuzzy clustering and probabilistic PCA mixture models," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 4, pp. 508–516, Aug. 2005.

[16] T. C. Glenn, "Context-dependent detection in hyperspectral imagery," Ph.D. dissertation, Dept. Comput. Inform. Sci. Eng., Univ. Florida, Gainesville, FL, USA, 2013.

[17] C. Robert, and G. Casella, *Monte Carlo Statistical Methods*, 2nd ed. New York, NY, USA: Springer, 2005.

[18] C. E. Rasmussen, "The infinite gaussian mixture model," in *Proc. Neural Inf. Process. Syst.*, 1999, vol. 12, pp. 554–560.

[19] T. Griffiths and Z. Ghahramani, "Infinite latent feature models and the indian buffet process," 2005.

[20] M. Isard and A. Blake, "Condensation—conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, pp. 5–28, 1998.

[21] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[22] P. Gader *et al.*, "MUUFL Gulfport hyperspectral and LiDAR airborne data set," Univ. Florida, Gainesville, FL, USA, Tech. Rep. REP-2013-570, Oct 2013.

[23] S. Theodoridis and K. Konstantinos, *Pattern Recognition*, 2nd ed. Amsterdam, The Netherlands: Elsevier, 2003.

[24] D. Anderson, A. Zare, and S. Price, "Comparing fuzzy, probabilistic, and possibilistic partitions using the earth mover's distance," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 4, pp. 766–775, Aug. 2013.

[25] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 8, pp. 841–847, Aug. 1991.

[26] R. Babuska, P. J. Van der Veen, and U. Kaymak, "Improved covariance estimation for Gustafson-Kessel clustering," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2002, vol. 2, pp. 1081–1085.

Taylor Glenn (S'10–M'13) received the Ph.D. degree from the University of Florida, Gainesville, FL, USA, in December 2013.

His research work has specialized in pattern recognition and machine learning and the application of these methods to sensor data, in particular ground-penetrating radar and electromagnetic induction data for buried landmine detection, and hyperspectral image analysis. Before receiving his graduate degree, he cofounded 2G Engineering LLC, a company specializing in embedded systems design and quick-turn manufacturing. He has since founded Precision Silver LLC, a company producing image and data analysis software with a focus on aerial imagery and precision agriculture applications.

Alina Zare (S'07–M'08–SM'13) received the Ph.D. degree from the University of Florida, Gainesville, FL, USA, in December 2008.

She teaches and conducts research in the area of pattern recognition and machine learning in the Department of Electrical and Computer Engineering, the University of Missouri, Columbia, MO, USA. Her research interests include hyperspectral image analysis, synthetic aperture sonar analysis, LIDAR data analysis, landmine and explosive object detection, sparsity promotion, and machine learning.

Dr. Zare received the prestigious National Science Foundation CAREER Award for her research on "Supervised Learning with Incomplete and Uncertain Data" and a recipient of the National Geospatial-Intelligence Agency New Investigator Program Award for her research on "Functions of Multiple Instances for Hyperspectral Analysis."

Paul Gader (M'86–SM'09–F'11) received the Ph.D. degree in mathematics for image-processing-related research from the University of Florida, Gainesville, FL, USA, in 1986.

He was a Senior Research Scientist with Honeywell, a Research Engineer and a Manager with the Environmental Research Institute of Michigan, and a Faculty Member with the University of Wisconsin, Oshkosh, the University of Missouri, Columbia, and the University of Florida, where he is currently a Professor and the Chair of Computer and Information Science and Engineering. He performed his first research in image processing in 1984 working on algorithms for detection of bridges in forward-looking infrared imagery as a Summer Student Fellow at Eglin Air Force Base. His dissertation research focused on algebraic methods for parallel image processing. He has since worked on a wide variety of theoretical and applied research problems including fast computing with linear algebra, mathematical morphology, fuzzy sets, Bayesian methods, handwriting recognition, automatic target recognition, biomedical image analysis, landmine detection, human geography, and hyperspectral and Light Detection and Ranging image analysis projects. He has published hundreds of refereed journal and conference papers.