SMAP Image Segmentation Software Package User's Manual

Charles A. Bouman
School of Electrical Engineering
Purdue University
West Lafayette IN 47906
bouman@ecn.purdue.edu
(317) 494-0340
http://ee.www.ecn.purdue.edu/~bouman/

Michael Shapiro
U.S. Army CERL
Office of GRASS Integration
(217) 384-8937
shapiro@zorro.cecer.army.mil
Version 1.3
April 3, 1996

Contents

1 Introduction

The sequential maximum a posteriori (SMAP) algorithm was developed to perform computationally efficient and high quality segmentation of multispectral or vector valued imagery. The algorithm uses multiscale segmentation techniques together with Bayesian models to incorporate spatial context into the segmentation process. The details of the smap segmentation method are described in the reference [1, 2] and reference [3] illustrates how the SMAP algorithm can improve classification accuracy in a target application.

The SMAP software package consists of two programs: clust and seg. The program clust is used to automatically estimate the model parameters of each image class from training data. The model used for each class is a Gaussian mixture model which can approximate any multivariate distribution. Once the class model parameters have been stored, then the seg program may use these parameters, together with the multispectral images to automatically generate a segmentation.

The main file directory contains the following subdirectories.

documentation - contains this manual and other documentation.

matlab - contains utilities for generating synthetic test data, and utilities for reading and writing pgm files from matlab.

cluster - This subdirectory contains the "clust" clustering program used to generate class model parameters from training data. This program is also generally useful as an unsupervised clustering program. The program clust is a C program for clustering data using the EM algorithm, and than computing the order using the Rissenen order identification criteria. This work has not yet been published.

segment - This subdirectory contains the "seg" segmentation program that implements the SMAP segmentation algorithm documented in the journal publication [2].

pgm - This subdirectory contains ANSII C subroutines for reading and writing pgm, ppm, and pbm image file formats. These routines are only provided so that users may write their own programs for manipulating the input and output images from the seg program.

2 Demos

The software package contains a simple segmentation example which may be easily ran. First compile the code for your machine using the makefiles in cluster/lib and segment/lib . The code is written to compile on a SUN sparc solaris machine, but it should also compile on other computers with appropriate modification of the makefiles.

The utilities in the subdirectory matlab have already been used to generate synthetic data for a segmentation example. Run the shell script cluster/EXAMPLE1 to cluster this synthetic training data and generate a parameter file for the SMAP segmentation program.

Next run the shell script in segment/EXAMPLE1 to segment the synthetic image. The resulting segmentation will be contained in the directory segment/output in the pgm image file format. All image inputs and outputs to the seg program are in pgm format.

The directory matlab also contains two of examples that run in matlab, and some utilities for reading and writing pgm files from matlab. The first shell file matlab/EXAMPLE1 generates the synthetic data for the segmentation example above, and the second shell file matlab/EXAMPLE2 generates data for a simple clustering example.

3 Training

Training is required in order to determine the multispectral data behavior for each class of interest. (The same software can of course be used to train and segment images for which each pixel contains a vector of data other then multispectral information.)

The clust program is used to perform this training step. Refer to Appendix C for more details on how the clust algorithm works, and Appendix A for details on how to run the clust program. In order to run the clust algorithm you must first create a data file for each class of interest. Each data file contains a series of vectors in ASCII floating point format and on separate lines. Each vector should be a sample from the multivariate distribution of the corresponding class. The clust program will use these data vectors to estimate the class distribution.

Clust will generate a params file which contains all the parameters of the Gaussian Mixture distribution. Appendix B contains information on the contents of the parameter file.

4 Segmenting

The seg program may be used to perform the SMAP segmentation. The details on how to run the seg program are contained in Appendix A. The seg program requires that you provide a params file. This file may either be generated using the clust program, or it may be generated manually. Refer to Appendix B for details about the contents of the params file, and refer to Appendix C for more information about the physical and mathematical meaning of the mixture parameters contained in the params file.

A Appendix: Program Manual Pages

seg

This program performs contextual image classification using sequential maximum a posteriori (SMAP) estimation.

SYNOPSIS

```
seg
     -help -- produce this help menu
     -i image -- pgm format image(s);
        May either be a single image, or may be numbered
        numbered image.1 image.2 etc.
     -x width  -- Input image width
     -y height -- Input image height
     -p params -- File containing spectral parameters.
     -o output -- Segmentation image in pgm format
     -v level  -- Verbose operation; level>=0
     -<algorithm> -- May be one of SMAP, Maxlike; Default SMAP
     -B blocksize -- Processing block size; Default 128
     -Classnum -- Output class number in stead of class index
```

DESCRIPTION

The seg program is used to segment multispectral images using a spectral class model known as a Gaussian mixture distribution. Since Gaussian mixture distributions include conventional multivariate Gaussian distributions, this pro- gram may also be used to segment multispectral images based on simple spectral mean and covariance parameters.

seg has two modes of operation. The first mode is the sequential maximum a posteriori (SMAP) mode [1, 2]. The SMAP segmentation algorithm attempts to improve segmentation accuracy by segmenting the image into regions rather than segmenting each pixel separately (see NOTES).

The second mode is the more conventional maximum likelihood (ML) classification which classifies each pixel separately, but requires somewhat less computation. This mode is selected with the -Maxlike flag (see below).

COMMAND LINE Arguments

-help (optional) generate help menu

-i image (required) input image file(s) in pgm format. May either be a single image, or may be numbered image.1 image.2 etc.

-x width (required) Number of columns in input image.

-y height (required) Number of rows in input image.

-o output (required) Output segmentation image in pgm format. The gray level of 1 represents class 1, gray level 2 represents class 2, etc. .

-v level (optional) Verbose operation; level≥0. Generates diagnostic information.

-Maxlike (0ptional) Use maximum likelihood estimation (instead of smap). Normal operation is to use SMAP estimation (see DESCRIPTION).

-p params (required) The params file contains the spectral signatures (i.e., the statistics) for the classes to be identified in the image. This signature file is produced by the program clust.

-Classnum (optional) Segmentation image is generated to contain the output class number in stead of class index. The output class number may be specified for each class in the parameters file.

-B blocksize (default 128) This option specifies the size of the "window" to be used when reading the image data.

This program was written to be nice about memory usage without influencing the resultant classification. This option allows the user to control how much memory is used. More memory may mean faster (or slower) operation depending on how much real memory your machine has and how much virtual memory the program uses.

The size of the submatrix used in segmenting the image has a principle function of controlling memory usage; however, it also can have a subtle effect on the quality of the segmentation in the smap mode. The smoothing parameters for the smap segmentation are estimated separately for each submatrix. Therefore, if the image has regions with qualitatively different behavior, (e.g. natural woodlands and man-made agricultural fields) it may be useful to use a submatrix small enough so that different smoothing parameters may be used for each distinctive region of the image.

The submatrix size has no effect on the performance of the ML segmentation method.

clust

This program is used to generate class parameter files for the seg program.

SYNOPSIS

clust #_subclasses info_file output_params_file

#_subclasses - maximum number of subclusters which can be used for any class. Each subcluster corresponds to a component of a Gaussian mixture distribution.

info_file - A file which contains the the following information:

```
<\# of classes>
<data vector length>
<class 1 data file name> <\# of data vectors in class 1>
<class 2 data file name> <\# of data vectors in class 2>
                .                           .
                .                           .
                .                           .
<last class data file name> <\# of data vectors in last class>
```

Each data file <class k data file name> contains a list of <# of data vectors in class k> data vectors in ASCII floating point format. Each data vector is of length <data vector length> and is on a single line of the file. Each vector is represents the components of the vector valued image at a single pixel and from a single class k.

output_params_file - An ASCII file which contains the parameters for the Gaussian mixture model used for each class.

DESCRIPTION

The clust program is used to determine the parameters of a spectral class model known as a Gaussian mixture distribution. The parameters may be estimated from a series of data vectors corresponding to training samples for each multispectral class. The mixture class parameters are stored as a class signature which can be used for subsequent segmentation of the multispectral image.

The Gaussian mixture class is a useful model because it can be used to describe the behavior of an information class which contains pixels with a variety of distinct spectral characteristics. For example, forest, grasslands or urban areas are examples of information classes that a user may wish to separate in an image. However, each of these information classes may contain subclasses each with its own distinctive spectral characteristic. For example, a forest may contain a variety of different tree species each with its own spectral behavior.

The objective of mixture classes is to improve segmentation performance by modeling each information class as a probabilistic mixture with a variety of subclasses. The mixture class model also removes the need to perform an initial unsupervised segmentation for the purposes of identifying these subclasses. However, if misclassified samples are

used in the training process, these erroneous samples may be grouped as a separate undesired subclass. Therefore, care should be taken to provided accurate training data.

The clust algorithm estimates both the number of distinct subclasses in each class, and the spectral mean and covariance for each subclass. The number of subclasses is estimated using Rissanen's minimum description length (MDL) criteria [5]. This criteria attempts to determine the number of subclasses which "best" describe the data. The approximate maximum likelihood estimates of the mean and covariance of the subclasses are computed using the expectation maximization (EM) algorithm [11, 9].

B Appendix: Example Parameter File

Figure 1 is an annotated example of a parameter file which must be supplied for the seg segmentation program. The parameter file may be generated automatically using the clust program, or it my by generated manually.

The parameter file specifies a model for each class to be segmented in the multispectral image. Each class model is specified in the form of a Gaussian mixture model. The subroutines read_sig.c and write_sig.c will read and write these parameters from the SigSet data structure defined in the file sig.h.

Below is an example of a parameter file containing two classes for a two band image. The first class is a simple multivariate Gaussian distribution with a single subclass. The second class is a second order mixture with two subclasses.

The comment lines are not allowed in the actual parameter file.

```
title: Data_set_name
nbands: 2     /* number of spectral bands */
class:        /* begin class specification */
 classnum: 0  /* class number 0 */
 classtitle:  /* class title (optional) */
 classtype: 0 /* class type (optional) */
 npixels: 0   /* number of pixels (optional) */
 subclass:    /* begin subclass specification */
  pi: 1.0     /* relative weight of subclass component */
  means: 127.0 128.0   /* vector of mean values */
  covar:               /* covariance matrix */
    64.0 0.0
    0.0  32.0
 endsubclass: /* end subclass specification */
endclass:     /* end class specification */
class:        /* begin class specification */
 classnum: 1  /* class number 0 */
 classtitle:  /* class title (optional) */
 classtype: 0 /* class type (optional) */
 npixels: 0   /* number of pixels (optional) */
 subclass:    /* begin subclass specification */
  pi: 0.25    /* relative weight of subclass component */
  means: 32.0 32.0     /* vector of mean values */
  covar:               /* covariance matrix */
    100.0 2.0
    2.0   100.0
 endsubclass: /* end subclass specification */
 subclass:    /* begin subclass specification */
  pi: 0.75    /* relative weight of subclass component */
  means: 64.0 64.0     /* vector of mean values */
  covar:               /* covariance matrix */
    50.0 0.0
    0.0   25.0
 endsubclass: /* end subclass specification */
endclass:     /* end subclass specification */
```

Figure 1: An example of a params file required to specify the Gaussian mixture model used for each class.

C Appendix: Automatic Parameter Estimation for Mixture Signatures

It is often desirable to segment multispectral images into regions corresponding to various information classes. For example, forest, grasslands or urban areas are examples of information classes that may be of interest. In practice, these information classes often contain subclasses each with their own distinctive spectral characteristic. The objective of mixture classes is to form a probabilistic class model composed of a number of spectral subclasses. Each subclass is than characterized by a set of parameters describing the mean and variation of the spectral components.

For each mixture class, it is necessary to determine the number of subclasses and the parameters of each subclasses. This can be done by using a representative sample of training data from each class and estimating the number of subclasses and their parameters from this data.

Specifically, let Y be an M dimensional random vector representing a multispectral pixel from the information class of interest. Let us assume that this information class has K subclasses. The the following parameters are required to completely specify the k^{th} subclass.

π_k - the probability that a pixel has subclass k.

μ_k - the M dimensional spectral mean vector for subclass k.

R_k - the $M \times M$ spectral covariance matrix for subclass k.

Furthermore, let K denote the number of subclasses, then we use the notation π, μ, and R to denote the parameter sets $\{\pi_k\}_{k=1}^K$, $\{\mu_k\}_{k=1}^K$, and $\{R_k\}_{k=1}^K$. The complete set of parameters for the information class are then given by K and $\theta = (\pi, \mu, R)$. Notice that the parameters are constrained in a variety of ways. In particular, K must be an integer greater than 0, $\pi_k \geq 0$ with $\sum_k \pi_k = 1$, and $det(R) \geq \epsilon$ where ϵ may be chosen depending on the application. We will denote the set of admissible θ for a K^{th} order model by $\Omega^{(K)}$.

Let $Y_1, Y_2, \cdots, Y_N$ be N multispectral pixels sampled from the information class of interest. Furthermore, assume that for each pixel Y_i the subclass of that pixel is given by the random variable X_n Of course, X_n is usually not known, but it will be useful for analyzing the problem. Then assuming that each subclass has a multivariate Gaussian distribution, the probability density function for the pixel Y_n given that $X_n = k$ is given by

$$p_{y_n|x_n}(y_n|k,\theta) = \frac{1}{(2\pi)^{M/2}} |R_k|^{-1/2} \exp\left\{-\frac{1}{2}(y_n - \mu_k)^t R_k^{-1}(y_n - \mu_k)\right\} .$$

However, we do not know the subclass X_n of each sample, so to compute the density function of Y_n given the parameter θ we must apply the definition of conditional probability and sum over k.

$$p_{y_n}(y_n|\theta) = \sum_{k=1}^{K} p_{y_n|x_n}(y_n|k,\theta)\pi_k$$

The log of the probability of the entire sequence $Y = \{Y_n\}_{n=1}^N$ is then given by

$$\log p_y(y|K,\theta) = \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} p_{y_n|x_n}(y_n|k,\theta)\pi_k\right) . \quad (1)$$

The objective is then to estimate the parameters K and $\theta \in \Omega^{(K)}$. The maximum likelihood (ML) estimate is a commonly used estimate with many desirable properties. It is given by

$$\hat{\theta}_{ML} = \arg \max_{\theta \in \Omega^{(K)}} \log p_y(y|K, \theta)$$

Unfortunately, the ML estimate of K is not well defined because the likelihood may always be made better by choosing a large number of subclusters. Intuitively, the log likelihood may always be increased by adding more subclasses since more subclasses may be used to more accurately fit the data.

This problem of estimating the order of a model is known as order identification, and has been studied by a variety of researchers. Methods for estimating model order generally tend to require the addition of a penalty term in the log likelihood to account for the over-fitting of high order models. One of the earliest approaches to order identification was suggested by Akaike [4], and requires the minimization of the so called AIC information criteria. The AIC criterion is given by

$$AIC(K, \theta) = -2 \log p_y(y|K, \theta) + 2L$$

where L is the number of continuously valued real numbers required to specify the parameter θ. In this application,

$$L = K \left(1 + M + \frac{(M+1)M}{2} \right) - 1 \ .$$

However, an important disadvantage of the AIC criteria for a number of problems is that the AIC does not lead to a consistent estimator [6]. This means that as the number of observations tends to infinity, the estimated value for K does not converge to the true value.

Alternatively, another criterion was suggested by Rissanen [5] called the minimum description length (MDL) estimator. This estimator works by attempting to find the model order which minimizes the number of bits that would be required to code both the data samples y_n and the parameter vector θ. While a direct implementation of the MDL estimator may depend on the particular coding method used, Rissanen develop an approximate expression for the estimate based on some assumptions and the minimization of the expression

$$MDL(K, \theta) = -\log p_y(y|K, \theta) + \frac{1}{2} L \log(NM) \ .$$

Notice that the major difference between the AIC and MDL criteria is the dependence of the penalty term on the total number of data values NM. In practice, this is important since otherwise more data will tend to result in over fitting of the model. In fact, it has been shown that for a large number of problems, the MDL criteria is a consistent estimator of model order [7, 8]. Unfortunately, the estimation of model order for mixture models does not fall into the class of problems for which the MDL criteria is known to be consistent. This is due to the fact that the solution to the mixture model problem always falls on a boundary of the constraint space, so the normal results on the asymptotic distribution of the ML estimate are no longer valid. An alternative method for order identification which is known to be consistent for mixture models is presented in [9]. However, this method is computationally expensive when the dimensionality of the data is high.

Our objective will be to minimize the MDL criterion given by

$$MDL(K,\theta) = -\sum_{n=1}^{N} \log\left(\sum_{i=1}^{K} p_{y_n|x_n}(y_n|k,\theta)\pi_k\right) + \frac{1}{2}L\log(NM) \ . \tag{2}$$

Direct minimization of $MDL(\theta)$ is difficult for a number of reasons. First, the logarithm term makes direct optimization with π, μ, and R difficult. Second, minimization with respect to K is complex since for each value of K a complete minimization with respect to π, μ, and R is required. If the subclass of each pixel, X_n, where known, then the estimation of π, μ, and R would be quite simple. Unfortunately, X_n is not available. However, the expectation-maximization (EM) algorithm has been developed to address exactly this type of "incomplete" data problem [10, 11].

Intuitively, the EM algorithm works by first classifying the pixels Y_n according to their subclass, and then re-estimating the subclass parameters based on this approximate classification. An essential point is that instead of the membership to each subclass being deterministic, the membership is represented using a "soft" probability. The process is started by assuming the the true parameter is given by $\theta^{(K,i)}$. We index $\theta^{(K,i)}$ by K and i because ultimately the EM algorithm will result in a iterative procedure for improving the MDL criterion. The probability that pixel y_n belongs to subclass k may then be computed using Bayes rule.

$$p_{x_n|y_n}(k|y_n,K,\theta^{(K,i)}) = \frac{p_{y_n|x_n}(y_n|k,\theta^{(K,i)})\pi_k}{\sum_{l=1}^{K} p_{y_n|x_n}(y_n|l,\theta^{(K,i)})\pi_l} \ .$$

Then using these "soft" subclass memberships we will then compute new spectral mean and covariance estimates for each subclass. We will denote these new estimates by $\bar{\pi}_k$, $\bar{\mu}_k$ and $\bar{R}_k$ where

$$\bar{N}_k = \sum_{n=1}^{N} p_{x_n|y_n}(k|y_n,K,\theta^{(K,i)}) \tag{3}$$

$$\bar{\pi}_k = \frac{\bar{N}_k}{N} \tag{4}$$

$$\bar{\mu}_k = \frac{1}{\bar{N}_k}\sum_{n=1}^{N} y_n p_{x_n|y_n}(k|y_n,K,\theta^{(K,i)}) \tag{5}$$

$$\bar{R}_k = \frac{1}{\bar{N}_k}\sum_{n=1}^{N} (y_n-\bar{\mu}_k)(y_n-\bar{\mu}_k)^t p_{x_n|y_n}(k|y_n,K,\theta^{(K,i)}) \tag{6}$$

In order to formally derive the EM algorithm update equations, we must first compute the following function

$$Q(K',\theta;K,\theta^{(i)}) = E\left[\log p_{y,x}(y,X|K',\theta)|Y=y,K,\theta^{(K,i)}\right] - \frac{1}{2}L\log(NM)$$

where Y and X are the sets of random variables $\{Y_n\}_{n=1}^N$ and $\{X_n\}_{n=1}^N$ respectively, and y and x are realizations of these random objects. The fundamental result of the EM algorithm which is proven in [10] is that

$$Q(K',\theta;K,\theta^{(i)}) > Q(K,\theta^{(i)};K,\theta^{(i)}) \Rightarrow MDL(K',\theta) < MDL(K,\theta^{(K,i)}) \ .$$

The objective of the EM algorithm is then to iteratively optimize with respect to K', θ until a local minimum of the MDL function is reached.

In order to derive expressions for the EM updates, we first compute a more explicit form for the function $Q(K, \theta; K, \theta^{(K,i)})$. Here we assume that $K' = K$, but we will treat the case when $K' < K$ later. The Q function may be expressed in the following form by substituting in for $\log p_{y,x}(y, x|\theta)$ and simplifying.

$$
\begin{aligned}
&Q(K, \theta; K, \theta^{(K,i)}) = \\
&\quad \sum_{k=1}^{K} \bar{N}_k \left\{ -\frac{1}{2} trace[\bar{R}_k R_k^{-1}] - \frac{1}{2}(\bar{\mu}_k - \mu_k)^t R_k^{-1}(\bar{\mu}_k - \mu_k) - \frac{M}{2}\log(2\pi) - \frac{1}{2}\log(|R_k|) + \log(\pi_k) \right\} \\
&\quad -\frac{1}{2} L \log(NM)
\end{aligned}
$$

where $\bar{N}_k$, $\bar{\mu}_k$, and $\bar{R}_k$ are as given in (3), (4), and (5).

We will first consider the maximization of $Q(K, \theta; K, \theta^{(K,i)})$ with respect to $\theta \in \Omega^{(K)}$. This maximization of Q may be done using Lagrange multipliers and results in the update equations

$$
\begin{aligned}
(\pi^{(K,i+1)}, \mu^{(K,i+1)}, R^{(K,i+1)}) &= \arg \max_{(\pi,\mu,R)\in\Omega^{(K)}} Q(K, \theta; K, \theta^{(K,i)}) && (7) \\
&= (\bar{\pi}, \bar{\mu}, \bar{R}) && (8)
\end{aligned}
$$

where $(\bar{\pi}, \bar{\mu}, \bar{R})$ may be computed using (3), (4), (5), and (6).

While (8) shows how to update the parameter θ, it does not show how to change the model order K. Our approach will be to start with a large number of clusters, and then sequentially decrement the value of K. To do this, we can perform the optimization of (8) while constraining the solution to have order $K - 1$. That is

$$
\max_{\theta\in\Omega^{(K-1)}} Q(K-1, \theta; K, \theta^{(K,i)}) \tag{9}
$$

An alternative way of thinking about the constraint $\theta \in \Omega^{(K-1)}$ is that it requires that two subclasses from $\theta \in \Omega^{(K)}$ have the same parameters. In other words, the optimization of (9) requires that for two subclasses, denoted by l and m, have the property

$$
\begin{aligned}
\pi_l &= \pi_m = \pi_{(l,m)} && (10) \\
\mu_l &= \mu_m = \mu_{(l,m)} \\
R_l &= R_m = R_{(l,m)} \ .
\end{aligned}
$$

Maximization of $Q(K, \theta; K, \theta^{(K,i)})$ subject to this constraint results in the following updates for $k \neq l$ or m.

$$
\begin{aligned}
\pi_k &= \bar{\pi}_k \\
\mu_k &= \bar{\mu}_k \\
R_k &= \bar{R}_k \ .
\end{aligned}
$$

Notice that these estimates are the same as in the previous case, but for $k = l$ or m

$$\pi_{(l,m)} = \frac{\bar{\pi}_l + \bar{\pi}_m}{2} \tag{11}$$

$$\mu_{(l,m)} = \frac{\bar{\pi}_l \bar{\mu}_l + \bar{\pi}_m \bar{\mu}_m}{\bar{\pi}_l + \bar{\pi}_m} \tag{12}$$

$$R_{(l,m)} = \frac{\bar{\pi}_l \left(\bar{R}_l + (\bar{\mu}_l - \mu_{(l,m)})(\bar{\mu}_l - \mu_{(l,m)})^t \right) + \bar{\pi}_m \left(\bar{R}_m + (\bar{\mu_m} - \mu_{(l,m)})(\bar{\mu_m} - \mu_{(l,m)})^t \right)}{\bar{\pi}_l + \bar{\pi}_m} \tag{13}$$

We will use the notation $\theta_{(l,m)}$ to denote the optimal solution to (9) under the constraint of (10), and we will denote the unconstrained optimum by θ^*. Using (11), (12) and (13), we may define a distance function with the form

$$\begin{aligned} & d(l,m) \\ &= Q(K, \theta^*; K, \theta^{(K,i)}) - Q(K, \theta_{(l,m)}; K, \theta^{(K,i)}) \\ &= N\bar{\pi}_l \left\{ -\frac{M}{2}(1 + \log(2\pi)) - \frac{1}{2}\log(|\bar{R}_l|) \right\} \\ &\quad + N\bar{\pi}_m \left\{ -\frac{M}{2}(1 + \log(2\pi)) - \frac{1}{2}\log(|\bar{R}_m|) \right\} \\ &\quad - 2N\pi_{(l,m)} \left\{ -\frac{M}{2}(1 + \log(2\pi)) - \frac{1}{2}\log(|R_{(l,m)}|) \right\} \\ &\quad - \left(1 + M + \frac{(M+1)M}{2} \right) \log(NM) \ . \end{aligned}$$

Simplification, leads to the final expression

$$d(l,m) = \frac{N\bar{\pi}_l}{2} \log \left(\frac{|R_{(l,m)}|}{|\bar{R}_l|} \right) + \frac{N\bar{\pi}_m}{2} \log \left(\frac{|R_{(l,m)}|}{|\bar{R}_m|} \right) - \left(1 + M + \frac{(M+1)M}{2} \right) \log(NM) \tag{14}$$

where $R_{(l,m)}$ is computed using (11), (12), and (13). With the function $d(l,m)$ precisely defined, it is now possible to search over the set of all pairs, (l,m), to find the cluster pair which minimizes $d(l,m)$ and therefore, maximizes the value of the Q function.

$$\theta^{(K-1,1)} = \theta_{(l^*,m^*)} \text{ where } (l^*, m^*) = \arg\min_{(l,m)} d(l,m) \ . \tag{15}$$

Notice that the new value of Q may either increase or decrease when the order is reduced. However, each reduction in order will be followed by EM updates of the parameters (π, μ, R) which may ultimately increase Q over the higher order model.

Before we can specify the final Cluster algorithm, we must specify the initial choice of the parameter $\theta^{(K,1)}$. The initial choice of $\theta^{(K,1)}$ can be important since the EM is only guaranteed to converge to converge to a local minimum. The initial number of subclusters, K_o, is chosen by the user subject to the constraint that the total number of parameters, $L < \frac{1}{2}MN$. The initial subclass parameters are then chosen to be

$$\pi_k^{(1)} = \frac{1}{K_o} \tag{16}$$

$$\mu_k^{(1)} = y_n \text{ where } n = \lfloor (k-1)(N-1)/(K_o - 1) \rfloor + 1 \tag{17}$$

$$R_k^{(1)} = \frac{1}{N} \sum_{n=1}^{N} y_n y_n^t \tag{18}$$

where $\lfloor \cdot \rfloor$ is the greatest smaller integer function.

The final Cluster algorithm is given in the following steps.

1. Initialize the class with a large number of subclasses, K_o.
2. Initialize $\theta^{(1)}$ using (16), (17) and (18).
3. Apply the iterative EM algorithm (8) until the change in $MDL(K, \theta)$ is less then ϵ.
4. Record the parameter $\theta^{(K, i_{final})}$, and value $MDL(K, \theta^{(K, i_{final})})$.
5. If the number of subclasses is greater than 1, apply equation (15) to reduce the number of clusters, set $K \leftarrow K - 1$, and go back to step 3.
6. Choose the value K^* and parameters $\theta^{(K^*, i_{final})}$ which minimize the value of MDL.

In step 3, the value of ϵ is chosen to be

$$\epsilon = \frac{1}{100}\left(1 + M + \frac{(M+1)M}{2}\right)\log(NM) \ .$$

References

[1] C. Bouman and M. Shapiro, "Multispectral Image Segmentation using a Multiscale Image Model," *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.,* pp. III-565 - III-568, San Francisco, California, March 23-26, 1992.

[2] C. A. Bouman and M. Shapiro, "A Multiscale Random Field Model for Bayesian Image Segmentation," *IEEE Trans. on Image Processing,* vol. 3, no. 2, pp. 162-177, March 1994.

[3] J. D. McCauley and B. A. Engel, "Comparison of Scene Segmentation SMAP, ECHO, and Maximum Likelihood" *IEEE Trans. on Geoscience and Remote Sensing,* vol. 33, no. 6, pp. 1313-1316, November 1995.

[4] H. Akaike, "A New Look at the Statistical Model Identification", *IEEE Trans. Automat. Contr.,* vol. AC-19, pp. 716-723, December 1974.

[5] J. Rissanen, "A Universal Prior for Integers and Estimation by Minimum Description Length," *Annals of Statistics,* vol. 11, no. 2, pp. 417-431, 1983.

[6] R. L. Kashyap, "Inconsistency of the AIC Rule for Estimating the Order of Autoregressive Models," *IEEE Trans. Automat. Contr.,* vol. AC-25, no. 5, Oct. 1980.

[7] R. L. Kashyap, "Optimal Choice of AR and MA Parts in Autoregressive Moving Average Models," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. PAMI-4, no. 2, pp. 99-104, March 1982.

[8] M. Wax and T. Kailath, "Detection of Signals by Information Theoretic Criteria," *IEEE Trans. on Acoustics, Speech, and Signal Processing,* vol. ASSP-33, no. 2, pp. 387-392, April 1985.

[9] E. Redner and H. Walker, "Mixture Densities, Maximum Likelihood and the EM Algorithm," *SIAM Review,* vol. 26, no. 2, April 1984.

[10] L. Baum, T. Petrie, G. Soules, N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *Ann. Math. Statistics,* vol. 41, no. 1, pp. 164-171, 1970.

[11] A. Dempster, N. Laird and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Roy. Statist. Soc. B,* vol. 39, no. 1, pp. 1-38, 1977.