

LA-UR- 97 - 2702

Approved for public release;
distribution is unlimited.

CONF-970706--

Title:

A Contiguity-Enhanced K-Means Clustering
Algorithm for Unsupervised Multispectral
Image Segmentation

Author(s):

James Theiler, NIS-2
Galen Gisler, NIS-2

RECEIVED

NOV 03 1997

OSTI

Submitted to:

SPIE Annual Meeting, July 28-Aug 1, San
Diego, CA and Proc SPIE 3159,
Algorithms, Devices, and Systems for
Optical Information Processing, B.
Javidi and D. Psaltis, editors.

MASTER

19980407 091

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Los Alamos
NATIONAL LABORATORY

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. The Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

A contiguity-enhanced k-means clustering algorithm for unsupervised multispectral image segmentation

James Theiler and Galen Gisler

Astrophysics and Radiation Measurements Group, NIS-2
Nonproliferation and International Security Division, MS-D436
Los Alamos National Laboratory, Los Alamos, NM 87545 USA

ABSTRACT

The recent and continuing construction of multi- and hyper-spectral imagers will provide detailed data cubes with information in both the spatial and spectral domain. This data shows great promise for remote sensing applications ranging from environmental and agricultural to national security interests. The reduction of this voluminous data to useful intermediate forms is necessary both for downlinking all those bits and for interpreting them. Smart on-board hardware is required, as well as sophisticated earth-bound processing.

A segmented image (in which the multispectral data in each pixel is classified into one of a small number of categories) is one kind of intermediate form which provides some measure of data compression. Traditional image segmentation algorithms treat pixels independently and cluster the pixels according only to their spectral information. This neglects the implicit spatial information that is available in the image.

We will suggest a simple approach — a variant of the standard k-means algorithm — which uses both spatial and spectral properties of the image. The segmented image has the property that pixels which are spatially contiguous are more likely to be in the same class than are random pairs of pixels. This property naturally comes at some cost in terms of the compactness of the clusters in the spectral domain, but we have found that the spatial contiguity and spectral compactness properties are nearly “orthogonal,” which means that we can make considerable improvements in the one with minimal loss in the other.

Keywords: algorithm, image segmentation, clustering, k-means

One might wonder why anyone is interested in such an unpromising problem, and whether or not it is even possible in principle to learn anything of value from unlabelled samples.

— Duda and Hart, *Pattern Classification and Scene Analysis*, 1973.

1. INTRODUCTION

Increasingly, in remote sensing of the earth and of other planets by orbiting satellites, and in observational astronomy, image data are acquired simultaneously in several distinct spectral bands. For example, Fig. 1 shows two scenes imaged by the Landsat Thematic Mapper in seven spectral bands. The number of spectral bands available on modern sensors is growing, and hyperspectral systems can have hundreds or more. The challenges involved in downlinking, reducing, analyzing, and interpreting such huge datacubes are considerable. Such datasets are rarely “labelled” with information about ground (or sky) truth, because such information is expensive to acquire, and difficult to register. One is led to ask: what *can* be learned from the unlabelled data?

In the context of multispectral imagery, a natural suggestion is to cluster the individual pixels into a small number of classes, each representing a different spectral type, and then to segment the image into those classes. The idea is that an image segmented by spectral category can be more informative than the image in any one of the individual spectral bands. This kind of segmentation is often used for preliminary or exploratory data analysis, because it provides a compression of detail, something that is increasingly important as the raw size of multispectral datacubes continues to grow. As well as data compression — both informally so that the human can interpret the information without being overwhelmed by it, and more formally so that hardware data storage and bandwidth requirements can be reduced — there are other motivations for clustering and segmentation of multispectral imagery.

Work supported by the U.S. Department of Energy, through the Deployable Adaptive Processing Systems (DAPS) project. Email: {jt,ggisler}@lanl.gov.

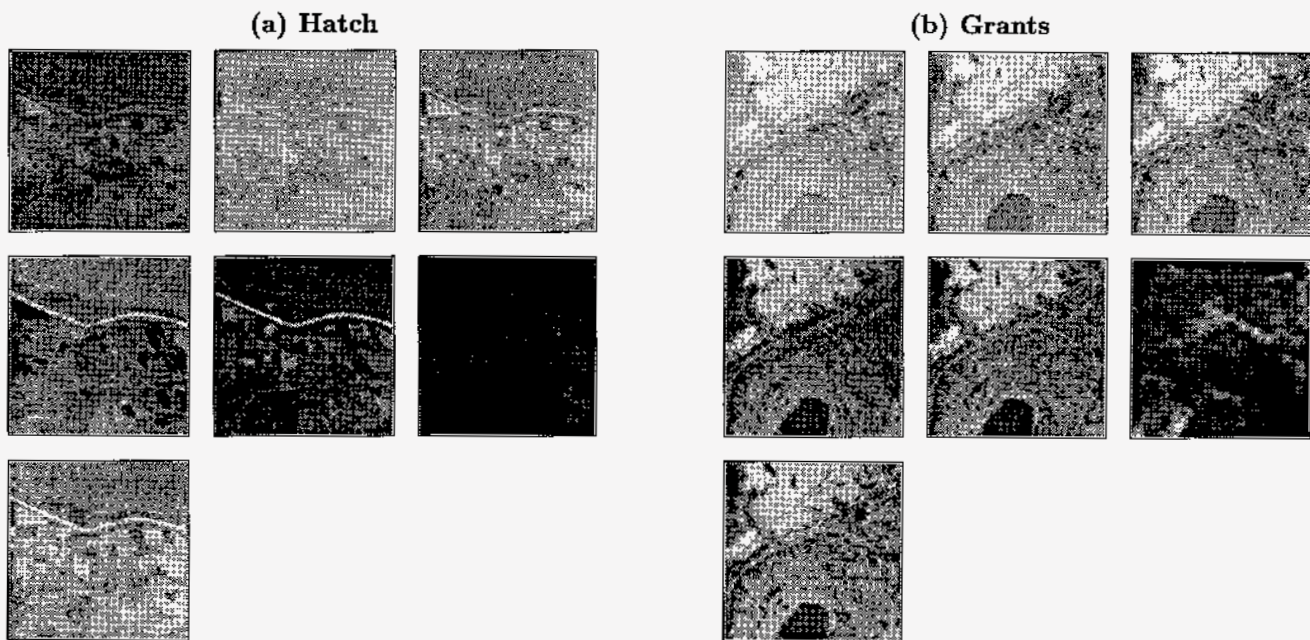


Figure 1. Two seven-band 151×151 pixel Landsat images, taken of scenes (a) near Hatch, New Mexico; and (b) near Grants, New Mexico. The spectral bands for the Landsat images are in the visible and infrared: $0.45\text{-}0.53\mu\text{m}$, $0.52\text{-}0.60\mu\text{m}$, $0.63\text{-}0.69\mu\text{m}$, $0.76\text{-}0.90\mu\text{m}$, $1.55\text{-}1.75\mu\text{m}$, $10.40\text{-}12.50\mu\text{m}$, and $2.08\text{-}2.35\mu\text{m}$. The lower resolution at the much longer wavelength is evident in the sixth channel.

- Combined with a small number of labelled samples, a large quantity of unlabelled samples can provide a way to “tune up” an algorithm for predicting the labels. The idea goes back at least two decades,¹⁻³ but good practical algorithms are still being developed.⁴ Castelli and Cover⁵ have argued that in the limit of many unlabelled samples, the labelled samples are “exponentially valuable.” This is good news to remote sensors, for it says that a little ground truth can go a long way.
- Clustering is also useful for nonstationary data. If the properties of labelled groups (or, of the sensor which is taking the data) change slowly over time, clustering can be used to follow those changes. Schowengerdt⁶ [pp. 196–202] for instance notes that the segmented images are less sensitive than the raw data to atmospheric conditions.
- For remote detection and characterization of gaseous plumes or land-based targets, the ground scene ceases to be the signal of interest, and becomes instead the clutter. Clustering may provide a way of reducing this background clutter, because the within-class variance of a segmented image can be much smaller than the overall variance of the image as a whole. The issues of clustering and pixel mixing are somewhat at odds with each other, but a recent paper by Stocker and Schaum⁷ points to one approach for combining them.

Our interest here is in “partitional” clusterings — these are single partitions of the data into disjoint classes. Hierarchical clusterings provide a “tree” of classes; the data is divided into clusters, those clusters are divided into subclusters, and so on. Fuzzy clusterings are single partitions of the data, but individual data points can be partial members of different classes.

2. SPATIO-SPECTRAL CLUSTERING

Standard cluster algorithms treat the objects they are clustering independently. Applied to multispectral data cubes, these algorithms (and many multispectral analysis methods) treat the pixels as if they were independent. This ignores *spatial* aspects, such as texture and contiguity, which for image data can potentially be very informative. If

one wants to take into account this spatial information, one must either alter the data representation so that each pixel is extended to include information about its neighbors, or else one must modify the algorithms themselves.

If one includes spatial information in the data representation, then one can use the old algorithms. A very general (and very expensive) way to account for the relationships of neighboring pixels is to embed the all the neighboring pixels into components of the pixel of interest. For instance, if we consider a 3×3 neighborhood of a 7-channel Landsat image, then the dimension of each enhanced pixel is $3 \times 3 \times 7 = 63$. Standard algorithms can then in principle be applied to these 63-dimensional objects, but usually dimension reduction techniques, such as principal components analysis, will be needed to reduce the dimensionality as a preprocessing step. Although this approach is in principle more general, it can in practice be more limiting because the rapid increase in dimensionality with neighborhood size constrains the neighborhood to be small. A more directed approach is to limit the number of added components by deciding beforehand what kind of spatial relationships to include. Texture-specific features, for instance, can be specified with convolutions of the image with spatial filters.⁸

On the other hand, it is also possible to modify the algorithms themselves, and continue to work on the data in its original lower-dimensional representation. This is the approach that we take in our "contiguity-enhanced" clustering algorithm because the information we desire from a neighboring pixel (namely, which class it belongs to) is not available from the image data itself.

3. OBJECTIVE MEASURES OF CLUSTER QUALITY

Formally, a clustering is a partition of a discrete set of objects into a smaller discrete set of classes. A "good" clustering is one where objects in the same class are more or less alike, and objects in different classes are in some sense different. It is important to make a distinction between clustering that seeks to distinguish qualitatively different types (ash versus birch), even without labelling those types, and clustering that segregates quantitative features (tall and thin versus short and fat). While the ambitious first goal often stands out as a kind of holy grail for clustering algorithms, the more pedestrian second goal can still be quite useful in a number of practical situations.

These goals are not mutually exclusive, but they are in our view different. With the first goal in mind, one implicitly assumes that the data have some underlying multimodal structure; if not, then it is "invalid" to be clustering the data at all. Indeed, Jain and Dubes⁹ devote an entire chapter of their book to various measures of cluster validity. When the goal is to identify qualitatively distinct types, then it is also important to determine the "correct" number of such types. Our approach in this paper, however, is more oriented toward the second goal. We seek measures to quantify "how good" a clustering is without attempting to judge whether it is "good enough." And we will generally take the number K of clusters as an input parameter to be specified by the user instead of an output value determined by the algorithm. We will only briefly comment on reasons for preferring some values of K over others.

3.1. Notation

- Let k index the clusters, and
- Let K be the total number of clusters.
- Let i index the samples (the "pixels" in this case), so that we can
- Let x_i be the data in the i 'th sample. Note that in general, x_i is a vector-valued quantity.
- Let $k(i)$ denote the cluster to which x_i belongs.
- Let I_k denote the set of i 's which belong to the k 'th cluster.
- Let n_k be the number of elements in I_k ; *i.e.*, the number of samples in the k 'th cluster, and
- Let $N = \sum_k n_k$ be the total number of samples.
- Let $c_k = \sum_{i \in I_k} x_i / n_k$ be the center of the k 'th cluster, and
- Let $c = \sum_k n_k c_k / N = \sum_i c_{k(i)} / N$ be the center of the entire data set.

3.2. External and Internal measures

An *external* measure of cluster quality compares the clustering obtained using the data x_i with external information (such as ground truth). For instance, x_i might measure radiance in a number of spectral bands, and the categories of interest might be different kinds of land use (urban, desert, cultivated, etc.), different species of dominant vegetation,

etc. Choosing a good external measure of cluster quality is obviously application-dependent, and in practice boils down to comparing the **similarity** of two clusterings: one from remotely sensed data and one from ground truth. Since the clusters are by definition unlabelled, the similarity measure should not depend on such labels — in other words, there is no *a priori* way to “match up” the clusters from the different clusterings. If the number K of clusters is even moderately large, the $[K(K - 1)/2]!$ possible combinations rules out trying them all. We have employed an approach suggested by by Rand,¹⁰ which is $O(N^2)$. This is also expensive, but it at least avoids the combinatorial explosion. Basically, all sample pairs (x_i, x_j) are considered. For a given sample pair, and for a given clustering, the two members of the pair are either in the same cluster or are in different clusters. If the pair members are in the same cluster for both clusterings, or if the pair members are in different clusters for both clusterings, then a running sum is incremented. This sum is divided by the total number of sample pairs to produce a measure of similarity that varies from 0 to 1, with identical clusterings achieving a similarity of 1.

But for remotely sensed data, it is usually difficult to obtain this kind of external information registered on a pixel-by-pixel basis. Therefore, we ask for *internal* measures of cluster quality. Jain and Dubes⁹ speak of these internal measures in the context of cluster validity, but we are not really concerned with whether or not our clusterings are “valid” in some absolute sense; instead, we are interested in comparing the relative “value” of clusterings.

The first and main such measure is the **compactness** of the individual clusters. One wants items within the same category to be as nearly identical as possible. A natural measure of (non)compactness is the average within-cluster variance. For a single cluster, the variance is given by

$$V_k = \frac{1}{n_k} \sum_{i \in I_k} (x_i - c_k)^2, \quad (1)$$

and averaging over the K clusters gives

$$V = \sum_{k=1}^K n_k V_k / N = \sum_{i=1}^N (x_i - c_{k(i)})^2 / N. \quad (2)$$

It is useful to normalize this value by the overall variance $V_o = \sum_{i=1}^N (x_i - c)^2 / N$ of the data to obtain the normalized mean squared error:

$$V^* = \frac{V}{V_o} = \frac{\sum_{i=1}^N (x_i - c_{k(i)})^2}{\sum_{i=1}^N (x_i - c)^2}. \quad (3)$$

This value varies from zero to one, with smaller values indicating better (*i.e.*, more compact) clusterings.

For a fixed number K of clusters, this measure of compactness provides a reasonable, objective, and (more or less) application-independent way to compare the quality of different clusterings. But one can always reduce the within-cluster variance by increasing K , all the way to the extreme limit where each sample is its own cluster. One can speak of an optimal clustering at a fixed K , but the compactness measure cannot by itself identify an optimal K .

A related, but more complicated measure, suggested by Coggins and Jain,⁸ defines the **isolation** of a single cluster as the ratio of the distance to the nearest cluster, divided by the the (rms) average radius of the cluster — this is a kind of “number of sigmas” to the nearest cluster:

$$S_k = \sqrt{\frac{\min_j (c_j - c_k)^2}{\sum_{i \in I_k} (x_i - c_k)^2 / n_k}}. \quad (4)$$

A large value suggests a compact well-isolated cluster. This measure was used to characterize texture-based image segmentations, and the authors reported an empirically determined threshold of 1.70, below which a cluster was not considered well-isolated.

For clustering that is used in image segmentation, there is also a premium — albeit a secondary one, compared to compactness — on **contiguity**. It is generally preferred that adjacent pixels be in the same category. There is an implicit assumption here that what we are looking at on the ground has a spatial decorrelation length that is much

larger than a pixel's field of view. We define the (dis)contiguity at a single pixel as the fraction of its neighbors that are not in the same class:

$$D_i = \frac{\text{Number of adjacent pixels } j \text{ for which } k(i) \neq k(j)}{\text{Number of adjacent pixels}} \quad (5)$$

The number of neighboring pixels for a two-dimensional image is either eight or four, depending on whether diagonal neighbors are included. We have found better performance when the diagonal neighbors are used, so our results are all based on eight neighboring pixels. The (dis)contiguity of a cluster is given by

$$D_k = \sum_{i \in I_k} D_i / n_k, \quad (6)$$

and finally the global measure of (dis)contiguity for the clustering is

$$D = \sum_{k=1}^K n_k D_k / N = \sum_{i=1}^N D_i / N. \quad (7)$$

This value D varies from zero to one, with smaller values again indicating better (in this case, more contiguous) clusterings.

4. K-MEANS ALGORITHM

There are several variants of the k -means clustering algorithm, but most variants involve an iterative scheme that operates over a fixed number of clusters, while attempting to satisfy the following properties:

1. Each class has a center which is the mean position of all the samples in that class.
2. Each sample is in the class whose center it is closest to.

One usually starts with an initial clustering,¹¹ and then loops through the samples, reassigning each to the cluster whose center it is closest to, and then recomputes the center locations. The algorithm continues until no more samples are reassigned. At that point, both properties will be satisfied, though the obtained clustering is not necessarily (or usually) the only partition of the data that satisfies these two properties.

The term k -means is attributed (by Gowda¹²) to MacQueen,¹³ whose first implementation involved only a single pass through the data. One of the earliest *iterative* implementations is attributed (by MacQueen¹⁴) to Forgy,¹⁵ though in that implementation (as well as some more recent ones¹⁶), the cluster centers were not recomputed until after a full pass through the data. But in fact it is straightforward (and computationally inexpensive) to recompute the centers each time a point is moved. Specifically, upon moving the point x_i from cluster j over to cluster k , one updates according to:

$$c_j \leftarrow (n_j c_j - x_i) / (n_j - 1) \quad (8)$$

$$c_k \leftarrow (n_k c_k + x_i) / (n_k + 1) \quad (9)$$

$$n_j \leftarrow n_j - 1 \quad (10)$$

$$n_k \leftarrow n_k + 1 \quad (11)$$

Recomputing centers on the fly also has the advantage that it prevents the formation of empty clusters.

The general effect of these moves is to produce more compact clusters. In fact, each time a point is reassigned to a new cluster, the overall (non)compactness measure V , as defined by the average within-cluster variance in Eq. (2), is reduced. In general, reassigning a point from the cluster j to cluster k leads to a change

$$\Delta V = \frac{n_k}{n_k + 1} (x_i - c_k)^2 - \frac{n_j}{n_j - 1} (x_i - c_j)^2. \quad (12)$$

where the variables n_k , c_k , n_j , and c_j correspond to their values *before* the point is actually moved. Note that this expression takes into account the fact that the centers change when a point is moved from one cluster to another.

Although moving a point to a cluster with a nearer center necessarily decreases V , it is also possible to decrease V with a move to a cluster whose center (before the move) is not the nearest. If we take as our goal to minimize the within-cluster variance (instead of the less stringent goal to satisfy the above two criteria), then it makes sense to use $\Delta V < 0$ as the condition for making a move. Späth¹⁷ calls this approach the “exchange” method, and the simpler rule that moves points to the nearest cluster center the “minimal distance” method. In both cases, every move reduces the within-cluster variance V , so the algorithm is “greedy” and subject to trapping in local minima; but on the other hand, there is no danger that the algorithm will get into an “infinite loop” moving the same point back and forth between clusters.

Note that although the computation of V is in general an $O(N)$ process, ΔV is computed in $O(1)$ time.¹⁸ And although the computation of cluster centers from scratch is an $O(n_k)$ process, the recomputation after adding or deleting a point is $O(1)$. Thus, although the k -means algorithm minimizes a global criterion, the computations at each step are local.

Our implementation also included (as an option) a trick suggested by Montolio *et al.*¹⁶ for speeding up the k -means algorithm by setting up a “safe” radius around each cluster center. This radius is half the distance to the nearest cluster center.¹⁹ If a point is within the safe radius, then it should not be moved, and one can avoid computing the K distances that are normally required to decide whether or not to move it. This can in principle be a real time saver, because after the first few iterations, most points have settled into their final clusters. We found that this approach did provide some speedup when the number of clusters K was small, but every time a point is moved from one cluster to another, two of the cluster centers change, and one has to recompute all of the cluster-cluster distances in order to obtain the new safe radii. On the other hand, if the k -means were implemented so that it recomputed centers only after a full pass through the data, and if the clusters are reasonably well isolated, then considerable speedup should be possible.

In our implementation, the number K of clusters is fixed. Much of the effort in the development of k -means-style clustering algorithms has been aimed at discovering good heuristics for choosing this K . The k -means algorithm is often the “inner loop” of such algorithms for which splitting and merging of clusters are permitted operations as well as moving samples from cluster to cluster.^{15,20-22} We remark that many algorithms which adaptively infer the appropriate K from the data still require an input variable which serves essentially the same purpose: a distance threshold, for instance, or a “vigilance” parameter.²³

4.1. Contig- k -means

Although the k -means algorithm was originally designed to minimize the average within-cluster variance of a clustering, it is often not difficult to modify the algorithm so that it optimizes other conditions. This kind of extension is especially useful if moving a point from one cluster to another leads to a change in the new global criterion which can be computed locally in $O(1)$ time.

The contig- k -means algorithm is such an algorithm. The idea, an early version of which was briefly described in Ref. 24, is to minimize a linear combination of (non)compactness and (dis)contiguity. This requires that the user specify a parameter λ to define the relative importance of these two properties. Now, the goal is to minimize

$$E = \lambda D + (1 - \lambda)V^*. \quad (13)$$

Since the (dis)contiguity D can be expressed as a sum of (dis)contiguities at each pixel, the effect on D of moving a point from cluster j to cluster k is easy to compute. Basically, one computes D_i before and after the move; the difference is divided by N and then multiplied by two (to account for the changes in contiguity at neighboring pixels) to produce ΔD .

5. ILLUSTRATION ON LANDSAT DATA

We used the multispectral Landsat images shown in Fig. 1 to illustrate and compare k -means and contig- k -means clustering. We specified $K = 4$ clusters, and for a range of values of the contiguity weight λ , applied the contig- k -means iterations until a stable clustering was achieved. Fig. 2 illustrates some of the segmentations that were found. These segmentations were generated by first clustering the data with $\lambda = 0$, and then using that clustering as an initial condition for $\lambda = 0.1$, and so on for increasing λ . This was found to produce better clusterings (smaller

values of the criterion in Eq. (13)) than those generated by a fixed λ from a random¹¹ starting condition. However, this approach did not work well for large λ , where trapping in local minima becomes a real problem. This is most obvious for the case $\lambda = 1$, where contiguity is to be optimized with complete disregard for the spectral properties. The optimal clustering here has all pixels in one cluster, giving $D = 0$ and $V^* = 1$.

The most notable feature of contig- k -means clustering is that we can make considerable gains in the contiguity of a clustering with virtually no loss to the compactness (up to a point). This is illustrated in Fig. 3. A simple alternative to contig- k -means for producing contiguity-enhanced clusterings is to spatially smooth the clusterings. Schowengerdt⁶ [pp. 187–190] describes several approaches for this. Perhaps the simplest, attributed to Goldberg *et al.*,²⁵ is to do a majority-rule smoothing of the clustering itself. The dashed lines in Fig. 3 illustrate the effect of successive smoothings with a 3×3 kernel. Again, one trades compactness for contiguity, though one has less control over how much of each is traded, and in general pays a higher cost in compactness for the same benefit in contiguity than one does for the contig- k -means algorithm. On the other hand, the simple smoothing algorithm performed better in the large $\lambda \approx 1$ regime, generating clusterings with much lower (dis)contiguity, though at the expense of much greater (non)compactness. If one were interested in this regime, further improvements would be possible by using the smoothed clusterings as initial conditions to the contig- k -means algorithm.

6. CLUSTERING AS DATA COMPRESSION

We have spoken of an image segmentation as a compression of information. We will make that statement more quantitative in this section, and treat the clustering quite literally as a data compression scheme. All the information in a pixel's multichannel spectrum is collapsed down to a single category label, essentially providing a vector-quantized compression.^{26,27} From this point of view, it makes sense to monitor the entropy of a clustering. For a general partition with fraction p_k of samples in the k 'th cluster, the ordinary Shannon entropy (in bits) is given by $S = -\sum_k p_k \log_2 p_k$, or

$$S = \log_2 N - \left[(1/N) \sum_{i=1}^N \log_2 n_{k(i)} \right]. \quad (14)$$

This describes the average number of bits per pixel required to specify which class each of the pixels is in.

Consider an image with d spectral channels, and b bits of precision in each channel. The image will nominally contain bd bits of information for each pixel.

Suppose we cluster the data into K distinct clusters, and for each pixel, instead of storing the full bd bits, store only the name of the cluster. If S is the entropy of the clustering, then we'll need an average of S bits per pixel to indicate which cluster a pixel belongs to. This is (lossy) compression of $bd - S$ bits per pixel.

If we also store the residual distance from pixel data values to the the cluster center, then we can achieve lossless compression. Or, if we store appropriately quantized residuals, we can obtain compression with an adjustable degree of loss. In the lossless case, we write V as the mean within-cluster variance of the clustering, and V_o as the variance of the full data set, so $\sqrt{V_o/V}$ is the average linear compression factor, and $d \log_2 \sqrt{V_o/V}$ is the number of fewer bits per pixel in the residual data compared to the original data. Thus we'll need an average of $bd - \frac{d}{2} \log_2 (V_o/V)$ bits to describe the residual distances to the same b bits of precision as in the original image.

There will also be bdK bits of overhead for the whole image. In the limit of large image size, however, this will be negligible. This (lossless) compression scheme will therefore save us

$$B = \frac{d}{2} \log_2 (V_o/V) - S \quad (15)$$

bits per pixel. The two terms in this expression represent a tradeoff between two competing desires. First, we want to make the average cluster size V as small as possible. Smaller residuals require fewer bits. But we also want to make the entropy S as small as possible. The solid lines in Fig. 4 show how this expression for B varies with number K of clusters, using ordinary k -means clustering on the Hatch and Grants data. Increasing K makes for smaller residuals V , but larger entropy S . The tradeoff favors large K , but the gain at large K is marginal. This suggests a criterion for choosing an appropriate K that makes no assumptions about underlying structure of the data. A literal

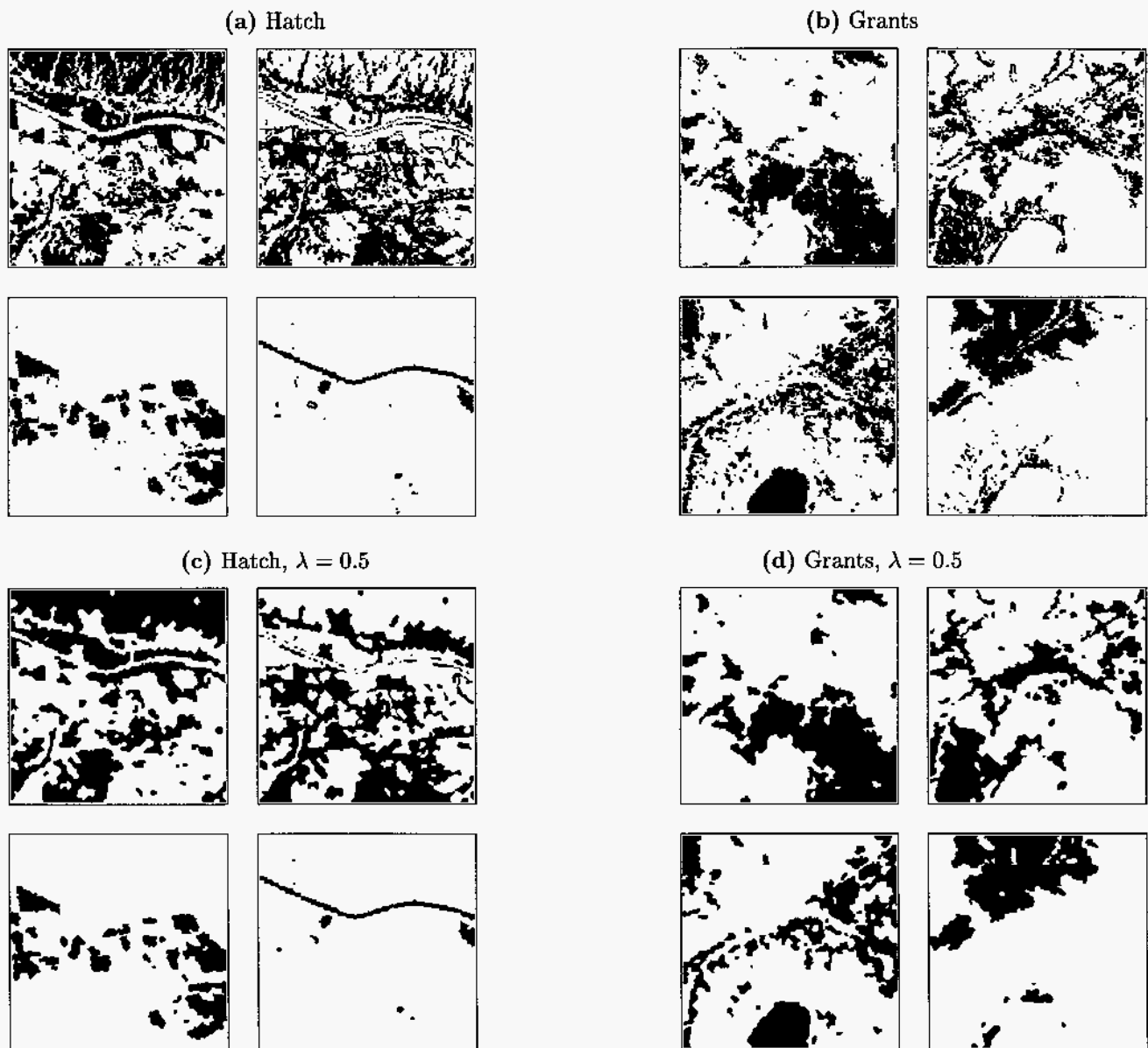


Figure 2. The top panels (a,b) show ordinary k -means clusterings ($\lambda = 0$) of the Landsat images shown in Fig. 1 into $K = 4$ distinct clusters. The bottom panels (c,d) show contig- k -means clusterings with $\lambda = 0.5$. The spatial contiguity is visibly enhanced. For the Hatch image (a,c), the (dis)contiguity measure decreased from 0.22 to 0.12, while the compactness changed by eight percent. The S_k statistic introduced by Coggins and Jain changed from 2.02 to 1.94; the Rand similarity statistic for clusterings (a) and (c) was 0.842. The numbers are similar for the Grants image (b,d): the (dis)contiguity again almost halved, going from 0.19 to 0.10, and the compactness again changed by only eight percent; the average S_k went from 2.04 to 1.96, and the similarity index was 0.912.

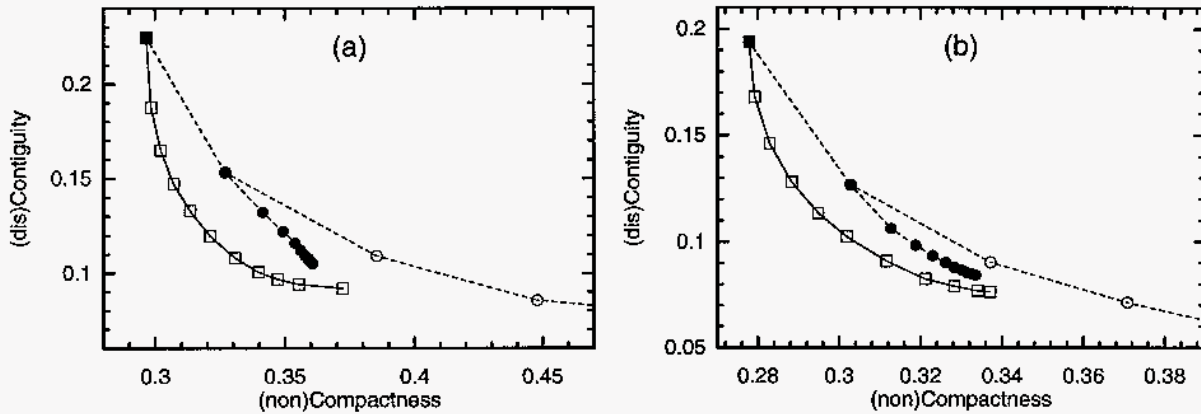


Figure 3. Plot of (dis)contiguity versus (non)compactness for clusterings generated for the multispectral data shown in Fig. 1: (a) Hatch; (b) Grants. The squares are from the *contig-k-means* algorithm using values of λ ranging from 0 (top, leftmost square) to 1.0 in steps of 0.1. Increasing values of λ led to clusters with smaller values for the (dis)contiguity and larger values for (non)compactness. For small values of λ , we can make considerable gains in the contiguity with little loss to the compactness. These curves trace out a kind of boundary below and to the left of which we do not expect any clusterings. The dotted lines correspond to smoothings of the $\lambda = 0$ clustering. The filled circles correspond to repeated application of a 3×3 kernel, and the open circles correspond to single application of kernels of size 3×3 , 5×5 , 7×7 , etc.

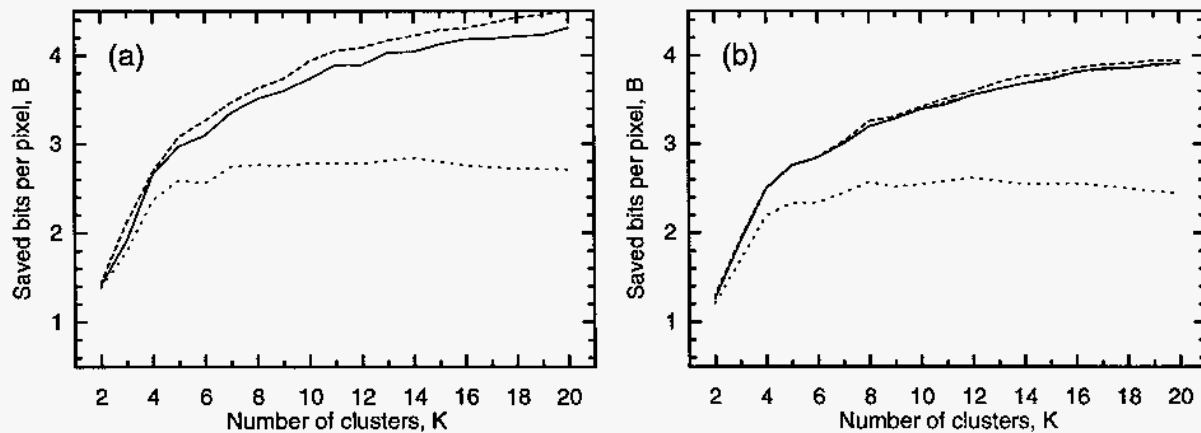


Figure 4. Plot of the number of bits per pixel that a lossless compression scheme would save as a function of the number K of clusters, based on (a) the Hatch image, and (b) the Grants image. The solid lines are based on ordinary *k-means* clustering with $\lambda = 0$. The dashed lines were obtained using a modification of *k-means* that explicitly maximizes the expression for bits per pixel of compression in Eq. (15). The dotted lines use *contig-k-means* with $\lambda = 0.5$.

optimization of K would have to consider the overhead bits as well as the effect of further compression, lossy or otherwise.

It is also possible to directly optimize the compression in Eq. (15) in a manner similar to that used by the contig- k -means algorithm to optimize the criterion in Eq. (13). To do this, we need to write a local expression for the effect on the global variable B due to moving a point from cluster j to cluster k .

$$\Delta B = \frac{d}{2} \Delta \log_2(V_o/V) - \Delta S = -\frac{d\Delta V}{2V} - \Delta S \quad (16)$$

where ΔV is given in Eq. (12), and ΔS can be derived from Eq. (14):

$$\Delta S = -\frac{1}{N} [(n_j - 1) \log_2(n_j - 1) - n_j \log_2 n_j + (n_k + 1) \log_2(n_k + 1) - n_k \log_2 n_k]. \quad (17)$$

The results of this algorithm are shown as dashed lines in Fig. 4. There is a slight improvement, but (in contrast to the case with contig- k -means) not a substantial one.

Note also that the flavor of this argument provides a more tangible justification for enhancing the "contiguity" of a clustering; a clustering with a high degree of contiguity can even more efficiently be compressed. The dotted lines in Fig. 4 show that using a contig- k -means clustering reduces the bits per pixel that the clustering method saves, but this may be made up for by a spatial compression stage that exploited the extra contiguity. Run-length encoding, for instance, should provide considerable compression for a highly contiguous segmented image.

If this kind of compression is going to take place on-board, then the clustering algorithms will have to take advantage of specialized hardware^{28,29} and/or parallel processing.^{30,31}

ACKNOWLEDGEMENTS

We are grateful to Chris Brislawn for discussion of data compression schemes and vector quantization; to Sophia Roberts for discussions of neural clustering algorithms; to Barry Smith for noting the relationship of clustering and clutter suppression and for pointing out Ref. 7; and to Pierre Villeneuve for comments on the manuscript.

REFERENCES

1. D. M. Titterton, "Updating a diagnostic system using unconfirmed cases," *Appl. Statist.* **25**, pp. 238-247, 1976.
2. T. J. O'Neil, "Normal discrimination with unclassified observations," *J. Amer. Statist. Assoc.* **73**, pp. 821-826, 1978.
3. G. J. McLachlan and S. Ganesalingam, "Updating the discriminant function on the basis of unclassified data," *Communications in Statistics - Simulation and Computation* **11**, pp. 753-767, 1982.
4. C. Ji and G. Boverman, personal communication.
5. V. Castelli and T. M. Cover, "On the exponential value of labeled samples," *Pattern Recognition Lett.* **16**, pp. 105-111, 1995.
6. R. A. Schowengerdt, *Techniques for Image Processing and Classification in Remote Sensing*, Academic Press, Orlando, 1983.
7. A. D. Stocker and A. P. Schaum, "Application of stochastic mixing models to hyperspectral detection problems," in *Algorithms for Multispectral and Hyperspectral Imagery III*, A. Iverson and S. S. Shen, eds., *Proc SPIE* **3071**, 1997.
8. J. M. Coggins and A. K. Jain, "A spatial filtering approach to texture analysis," *Pattern Recognition Lett.* **3**, pp. 195-203, 1985.
9. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, NJ, 1988.
10. W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Amer. Statist. Assoc.* **66**, p. 846, 1971.
11. The results we show here are based on an initial segmentation in which the i 'th vertical stripe of the image is assigned to class $i \bmod K$. This provides a modicum of initial contiguity ($D = 0.5$), without too strongly biasing the cluster formation.

12. K. C. Gowda, "A feature reduction and unsupervised classification algorithm for multispectral data," *Pattern Recognition* **17**, pp. 667–676, 1984.
13. J. MacQueen, "On convergence of k -means and partitions with minimum average variance," *Ann. Math. Statist.* **36**, p. 1084, 1965. (Abstract).
14. J. MacQueen, "Some methods of classification and analysis of multivariate observations," in *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, L. M. L. Cam and J. Neyman, eds., pp. 281–297, University of California Press, (Berkeley), 1967.
15. E. Forgy, "Cluster analysis of multivariate data: efficiency versus interpretability of classifications," *Biometrics* **21**, p. 768, 1965. (Abstract).
16. P. Montolio, A. Gasull, E. Monte, L. Torres, and F. Marqués, "Analysis and optimization of the k -means algorithm for remote sensing applications," in *Pattern Recognition and Image Analysis*, N. P. de la Blanca, A. Sanfeliu, and E. Vidal, eds., vol. 1 of *Series in Machine Perception and Artificial Intelligence*, pp. 155–170, World Scientific, (Singapore), 1992.
17. H. Späth, *Cluster Dissection and Analysis: Theory, FORTRAN programs, Examples*, Ellis Horwood Limited, Chichester, UK, 1985.
18. Actually, the computation requires $O(d)$ time where d is the number of dimensions, but we are treating d as if it were of order one — treating, but not assuming $d = O(1)$, since d multiplies all the other times as well, so the comparisons between them are all still valid.
19. If one wants to move points according to the $\Delta V < 0$ criterion, instead of the nearest center criterion, then the safe radius is slightly less than half the distance to the next cluster. If n_j is the number of samples in the current cluster, and n_k the the number in the nearby cluster, then the safe radius is given by $(\sqrt{1 + \epsilon} - 1)/\epsilon \approx 1/2 - \epsilon/8$ where $\epsilon = (n_j + n_k)/[n_k(n_j - 1)]$.
20. G. H. Bull and D. J. Hall, "ISODATA – a novel method of data analysis and pattern classification," tech. rep., Stanford Research Institute, Menlo Park, CA, 1965.
21. F. R. Fromm and R. A. Northhouse, "Class: a nonparametric clustering algorithm," *Pattern Recognition Lett.* **8**, pp. 107–114, 1976.
22. R. Dubes and A. K. Jain, "Clustering techniques: the user's dilemma," *Pattern Recognition Lett.* **8**, pp. 247–260, 1976.
23. G. A. Carpenter and S. Grossberg, "The ART of adaptive pattern recognition by a self-organizing neural network," *Computer* **21**, pp. 77–88, 1988.
24. S. Roberts, G. Gisler, and J. Theiler, "Spatio-spectral image analysis using classical and neural algorithms," in *Smart Engineering Systems: Neural Networks, Fuzzy Logic, and Evolutionary Programming*, C. H. Dagli, M. Akay, C. L. P. Chen, B. R. Fernández, and J. Ghosh, eds., vol. 6 of *Intelligent Engineering Systems Through Artificial Neural Networks*, pp. 425–430, ASME Press, (New York), 1996.
25. M. Goldberg, D. Goodenough, and S. Shlien, "Classification methods and error estimation for multispectral scanner data," in *Proc. Third Canadian Symposium on Remote Sensing*, pp. 125–143, 1975.
26. R. M. Gray, "Vector quantization," *IEEE ASSP Mag.* **1**, pp. 4–29, 1984.
27. W. H. Equitz, "A new vector quantization clustering algorithm," *IEEE Trans. Acoust. Speech, Signal Processing* **37**, pp. 1568–1575, 1989.
28. L. M. Ni and A. K. Jain, "A VLSI systolic architecture for pattern clustering," *IEEE Trans. on Pattern Anal. and Mach. Intel. PAMI* **7**, pp. 80–89, 1985.
29. E. L. Zapata, R. Doallo, F. F. Rivera, and M. A. Ismail, "A VLSI systolic architecture for fuzzy clustering," *Microprocessing and Microprogramming* **24**, pp. 647–654, 1988.
30. J. C. Tilton and J. P. Strong, "Analyzing remotely sensed data on the massively parallel processor," in *Proceedings Seventh International Conference on Pattern Recognition*, (Montreal), 1984.
31. F. F. Rivera, M. A. Ismail, and E. L. Zapata, "Parallel squared error clustering on hypercube arrays," *J. Parallel and Distributed Computing* **8**, pp. 292–299, 1990.

M98000818



Report Number (14) LA-UR--97-2702
CONF-970706--

Publ. Date (11) 199710
Sponsor Code (18) DOE/DP, XF
UC Category (19) UC-700, DDE/ER

DOE