Design Issues for Hardware Implementation of an Algorithm for Segmenting Hyperspectral Imagery

James Theiler[1], Miriam Leeser[2], Michael Estlick[2], and John J. Szymanski[1]

[1]Space and Remote Sensing Sciences Group
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

[2]Department of Electrical and Computer Engineering
Northeastern University
Boston, Massachusetts 02115

ABSTRACT

Modern hyperspectral imagers can produce data cubes with hundreds of spectral channels and millions of pixels. One way to cope with this massive volume is to organize the data so that pixels with similar spectral content are clustered together in the same category. This provides both a compression of the data and a segmentation of the image that can be useful for other image processing tasks downstream.

The classic approach for segmentation of multidimensional data is the k-means algorithm; this is an iterative method that produces successively better segmentations. It is a simple algorithm, but the computational expense can be considerable, particularly for clustering large hyperspectral images into many categories. The ASAPP (Accelerating Segmentation And Pixel Purity) project aims to relieve this processing bottleneck by putting the k-means algorithm into field-programmable gate array (FPGA) hardware.

The standard software implementation of k-means uses floating-point arithmetic and Euclidean distances. By fixing the precision of the computation and by employing alternative distance metrics (we consider the "Manhattan" and the "Max" metrics as well as a linear combination of the two), we can fit more distance-computation nodes on the chip, obtain a higher degree of fine-grain parallelism, and therefore faster performance, but at the price of slightly less optimal clusters. We investigate the effects of different distance metrics from both a theoretical (using random simulated data) and an empirical viewpoint (using 224-channel AVIRIS images and 10-channel multispectral images that are derived from the AVIRIS data to simulate MTI data).

Keywords: hyperspectral, k-means, image segmentation, image processing, field-programmable gate array (FPGA)

1. INTRODUCTION

Hyperspectral images, with detailed spatial resolution and upwards of 200 spectral channels, are becoming an increasingly common data product in remote sensing applications. These data provide important and useful information, but the challenge for the data analyst is to identify the important and useful features without being overwhelmed by the sheer volume of the data. One approach is provided by algorithms which segment the image by clustering pixels into classes, based on the spectral similarity of each pixel to other members of the class. This can represent a massive, albeit lossy, compression of data. For instance, we have been working with AVIRIS data,[1] in which each 614x512 pixel image has 224 16-bit channels; about 140 MB of data. When clustered into 16 classes, each pixel is represented by a 4-bit pointer to its class. The resulting image is about 150KB – a compression factor of three orders of magnitude.

As well as reducing the data for quicklook views, clustering also provides an organization of the data that can be useful for other processing downstream. Several authors have shown that clustering the data beforehand increases the performance of algorithms which attempt to "learn" features from a small number of examples.[2,3] Schowengerdt[4] suggests the use of image segmentation for change detection: a change in the segmentation is more likely to indicate an actual change on the ground, since the segmentation is relatively robust to changes in sensor performance and atmospheric conditions. It has also been argued that signal-to-clutter ratios can be improved by treating individual clusters separately; the variance within a cluster is generally much smaller than the variance over the whole image.

While there are clear benefits to clustering high-dimensional data sets, workstation implementations of clustering algorithms can be slow. Most algorithms, such as k-means, are iterative and require numerous passes through the data before convergence is achieved. Each iteration requires a computation of distance from every data point to every cluster center, and each distance requires accessing all the spectral channels. To accelerate this computation, we are developing an implementation of the k-means algorithm on reconfigurable hardware, using FPGA (Field Programmable Gate Array) chips. Reconfigurable hardware provides the ability to exploit the fine-grained parallelism inherent in the computation, while maintaining the flexibility to consider variants on the design and on the algorithm. FPGAs are particularly well suited to this application because the amount of parallelism and processing element bit widths can adapt to the task, allowing the designer to take maximum advantage of the hardware at hand. In addition, clustering on an FPGA board frees the workstation for the analyst to apply other algorithms to the data.

Implementations in hardware involve a different set of design tradeoffs than implementations in software. For example, the main goal of a software implementation may be to minimize the number of iterations. In hardware, our goals are to simplify the underlying operations as much as possible in order to speed up the calculations and to be able to provide more parallelism. A simpler operation translates to a smaller area datapath which in turn translates to more copies replicated on the chip. To this end, we are investigating alternative distance measures. By using a Manhattan or a Max distance, instead of the more expensive Euclidean distance, we trade theoretical optimality for practical efficiency. The Manhattan and Max distance calculations both eliminate a multiplier, thus saving area. They also reduce the number of bits needed to store intermediate calculations, minimizing the area required for routing.

This paper describes a set of experiments to compare Euclidean and alternative metrics for assigning data to a cluster. We conduct two sets of experiments. The first is a set of data-independent experiments that show an idealized setup for estimating misclassification rates and their effect on the total within-class variance. The second set of experiments shows how the use of alternative distance metrics affects total variance for some AVIRIS[1] and simulated MTI[5] data sets.

2. K-MEANS CLUSTERING

Given a set of N pixels, each composed of D spectral channels, and represented as a point in D-dimensional Euclidean space (that is, $\mathbf{x}_n \in \mathcal{R}^D$, with $n = 1, \ldots, N$); we partition the pixels into K clusters with the property that pixels in the same cluster are spectrally similar. Each cluster is associated with a "prototype" or "center" value which is representative of (and close to) the pixels in that class.

One measure of the quality of a partition is the within-class variance; this is the sum of squared (Euclidean) distances from each pixel to that pixel's cluster center.

For a fixed partition, the optimal (in this sense of minimum within-class variance) location for each center is the mean of all pixels in each class. And for a fixed choice of centers, the optimal partition assigns each point to the cluster whose center is closest. The k-means clustering algorithms (there are several variants) provide an iterative scheme that operates over a fixed number (K) of clusters, while attempting to simultaneously optimize center locations and pixels assignments.

From an initial sampling, the algorithm loops over all the data points, and reassigns each to the cluster whose center it is closest to. After a full pass through the data, the cluster centers are recomputed. Note that other variants of k-means update the cluster centers each time a point is reassigned to a new cluster. This leads to faster convergence, but is more difficult to implement in hardware. Each iteration reduces the total within-class variance for the clustering, so it is guaranteed that after enough iterations, the algorithm will converge, and further passes will not reassign points. It bears remarking that this is a local minimum; and it bears further remarking that the final convergence can depend somewhat sensitively on the initialization of the algorithm.

3. ALTERNATIVE DISTANCE MEASURES

Points are assigned to the cluster centers to which they are closest; for the minimum-variance criterion, "closest" is defined in terms of the Euclidean distance. Consider a point $\mathbf{x}$ and cluster center $\mathbf{c}$ where i indexes the spectral components of each. The Euclidean distance is defined as:

$$\|\mathbf{x} - \mathbf{c}\|^2 = \sum_i |x_i - c_i|^2. \tag{1}$$

But other distance measures can also be used; for instance, the general family of p-metrics (for which the Euclidean distance is the special case $p = 2$) is given by:

$$\|\mathbf{x} - \mathbf{c}\|^p = \sum_i |x_i - c_i|^p. \tag{2}$$

To perform a k-means iteration, one must compute the distance from every point to every center. If there are N points, K centers, and D spectral channels, then there will be $O(NKD)$ operations. For the Euclidean distance, each operation requires computing the square of a number.

The Euclidean distance has several advantages. For one, the distance is rotationally invariant. Furthermore, minimizing the Euclidean distance minimizes the within-class variance. On the other hand, the Euclidean distance is more expensive than the alternatives that we are considering. The Manhattan distance, corresponding to $p = 1$, is the sum of absolute values of the coordinate differences; the Max distance, corresponding to $p = \infty$ is the maximum of the absolute values of the coordinate differences.

Neither of the alternative distances requires any multiplication, and the Max distance has the slight advantage that the number of bits required to express the total distance does not increase with the dimension D (for the Manhattan distance, we require $\log_2 D$ extra bits for the total distance over and above the bits required for the coordinate differences). Note that the Euclidean distance requires D extra bits (unless a square root is invoked!); so it is not only more computationally demanding, it also requires a wider data path than the alternative distance metrics.

In assigning a data point to a cluster center, we assign to the center which is closest according to one of the computationally cheaper metrics. We expect that in general closeness in the Manhattan or Max metric implies closeness in the Euclidean metric, but we recognize that this is an approximation that may lead to non-optimal clusterings. See Figure 1.

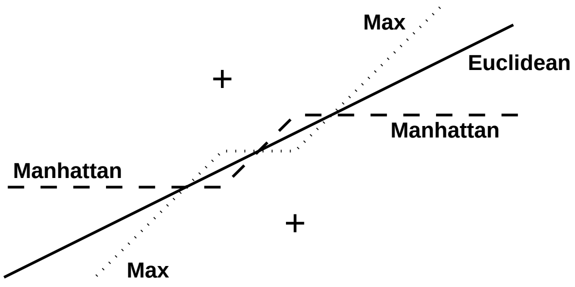

Figure 1. Two cluster centers are indicated by the two crosses in the figure above. The solid line partitions the area into points closer to the top cross and those closer to the bottom cross, where closeness is defined in terms of the Euclidean distance metric. If distance is defined in terms of a Manhattan metric, then the partition is given by the dashed line; if distance is given by the Max metric, then the partition is given by the dotted line. Points in the figure that are above the solid line, but below the dashed line will be misclassified by the Manhattan metric as belonging to the bottom class, when in fact they would be assigned by the Euclidean metric to the top class.

Our third alternative takes advantage of the fact that these two distances have values of p on either side of the $p = 2$ that corresponds to the Euclidean distance. By taking a linear combination of the Manhattan and the

Max distances, we can approximate the Euclidean distance more accurately than either of the two distances can individually. Further, the linear combination is straightforward to implement in hardware, and does not require any extra bit-width. The linear combination is given by

$$\|\mathbf{x} - \mathbf{c}\| = \alpha \max |x_i - c_i| + (1 - \alpha) \sum_{i=1}^{D} |x_i - c_i|. \tag{3}$$

The use of this linear combination was inspired by a pair of papers by Filip,[6,7] looking at linear and piecewise linear estimates of $\sqrt{x^2 + y^2}$ as a function of $x \leq y$. But where Filip's attention was restricted to two dimensions, our linear combination applies in arbitrary dimension; also, while Filip was interested in actual approximations, we are happy to approximate any monotonic function of the Euclidean distance, since our interest is in identifying which of two (or more) points is nearest to a given point.

4. DATA-INDEPENDENT ASSESSMENT OF ALTERNATIVE DISTANCE METRICS

In assessing this tradeoff, we performed an experiment to estimate how often points would be mis-assigned because a cheaper distance metric was used. The effect of metric choice on the quality of a clustering depends on many factors: the number of clusters, the dimension of the space, and the nature of the data in that space. To eliminate as many of these factors as possible, we considered an idealized situation in which three points are placed at random on the surface of a D-dimensional sphere (so that there are no "edge effects"). Two of the points are taken to be cluster centers, and the third is a data point. We compute the Euclidean distances from the data point to the two centers in order to determine which is truly closer. Then we compute the distances from the data point to the two centers using an alternative metric (Manhattan, Max, or linear combination thereof), and note whether this second pair of distances correctly identified the closest center. From a set of 10^5 such trials, we compute two statistics: relative variance and misclassification rate. The relative variance is the ratio of within-class variance for the cluster assignments provided by the alternative metric, divided by the within-class variance that would have been obtained if Euclidean distances had been used to assign points to centers. That the value is always larger than one reflects the fact that the Euclidean distance is the optimal choice for minimizing within-class variance. We estimated the misclassification rate by counting the fraction of trials for which the cheaper metric assigned the point incorrectly.

The results are shown in Fig. 2 as a function of the linear parameter α. Here $\alpha = 0$ corresponds to a pure Max metric, and $\alpha = 1$ corresponds to the Manhattan metric. The linear combination provides distances that are better than both the Max and Manhattan distances, and based on these results, we adopt $\alpha = 0.25$ as our "standard" value; it is near optimal over a wide range of dimensions D, and has the further advantage, for hardware implementation, that its denominator is a small power of two.

In Fig. 3, we plot the relative variance and misclassification rate as a function of the number D of spectral channels. As the dimension D increases, the mis-classification rate also increases. For the Manhattan metric (and for the $\alpha = 0.25$ linear-combination metric), this rate saturates at about fifteen percent, but for the Max metric, the error rate begins to approach fifty percent. That is, for very large dimension D, the Max metric is not much better than just assigning points to clusters at random.

We also performed this experiment for $K = 16$ classes. In this case, a trial consisted of 17 points placed at random on a D-dimensional sphere. Sixteen of the points were considered cluster centers, and the seventeenth point was the pixel whose class was assigned according to which of the 16 centers it was closest. The results of a Monte-Carlo experiment with 10^5 trials are shown in Fig. 4. The results are qualitatively similar to the $K = 2$ case, with the relative variance and the misclassification rate both larger in the large K case. Again, as D increases, the Max metric appears to be approaching a misclassification rate of 15/16 (that is, random assignment of centers to points), but the Manhattan and linear combination both saturate at around 40%; this is a pretty high rate of misclassification, but it is reasonable to presume that most of the misclassification are to classes that are nearby the true class.

5. USING REAL HYPERSPECTRAL AND MULTISPECTRAL DATA

The data-independent experiments provide some guidance with regard to the relative ability of the alternative distance metrics to classify high-dimensional data sets, but the simplified model misses some aspects of a full k-means clustering of a real hyperspectral dataset. For instance, one can presume that most of the misclassifications

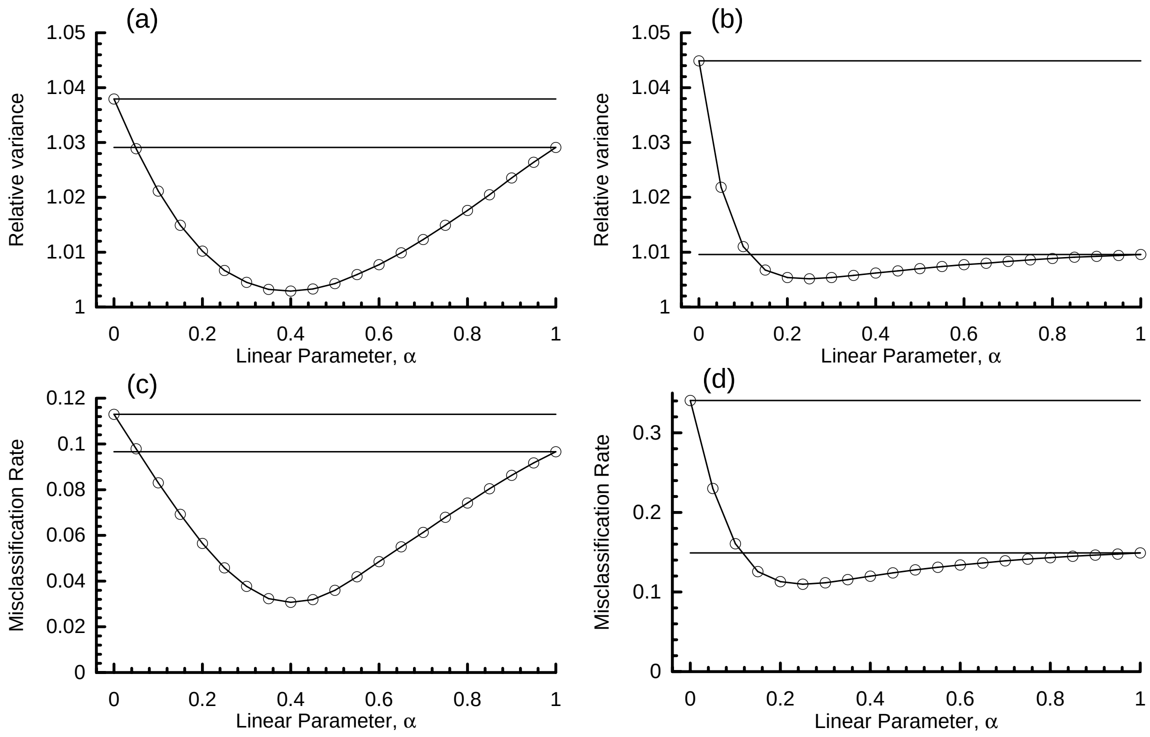

Figure 2. **(a)** This plot shows the relative variance for $K = 2$ classes, plotted against the parameter α of the linear combination of Max and Manhattan metrics (see Eq. (3)), for a low ($D = 3$) dimensional space. **(b)** Same as (a), but with a high ($D = 50$) dimensional space. The horizontal lines correspond to pure Max (upper line, corresponding to $\alpha = 0$) and pure Manhattan (lower line, corresponding to $\alpha = 1$). For the smaller value of D, the optimal linear combination provides a sizeable reduction in the relative variance, compared to the Manhattan and the Max metrics by themselves. For larger D, the linear combination is only slightly better than the Manhattan metric. The optimal value of α decreases, slowly, with increasing D; for this range of D, a value of $\alpha = 0.25$ works pretty well. **(c)** The probability of misclassification using the alternative metrics is shown as a function of α for $D = 3$. **(d)** Same as (c), but with $D = 50$. Particularly for this larger dimension, a small relative variance (less than a percent difference between the Euclidean and the $\alpha = 0.25$ alternative) can lead to a large misclassification rate (near ten percent).

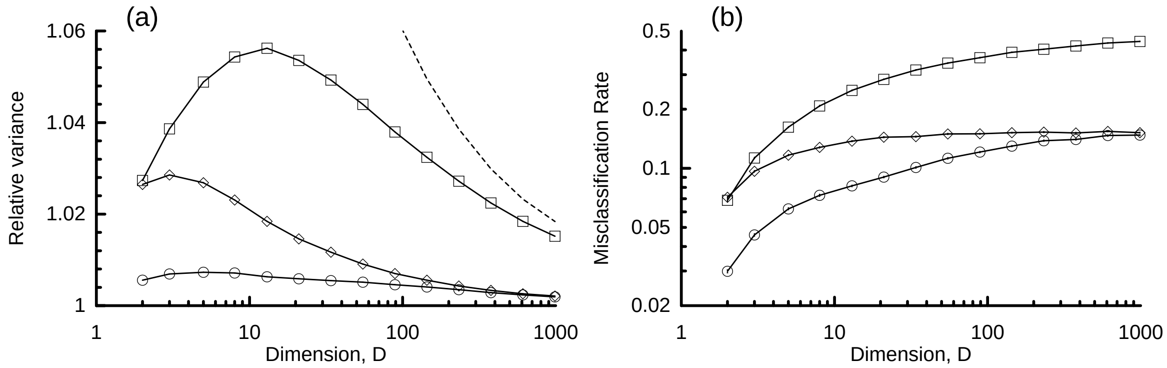

Figure 3. **(a)** Relative variance for $K = 2$ classes, plotted against the dimension D of the sphere onto which a triplet of points have been randomly placed. Two of the points in each triplet represent cluster centers, and the third a random pixel. The diamonds represent the Manhattan metric, the squares represent the Max metric, and the circles correspond to the linear combination with $\alpha = 0.25$. The dashed line indicates the relative variance associated with random assignment of pixels to centers. This is a "worst case" bound for the relative variance. **(b)** Probability of misclassification for the same parameters as in (a). A misclassification occurs when the center that is closest according to the Euclidean metric is not closest according to one of the other metrics. Note that although the relative variance is decreasing for large values of D, the rate of misclassification is monotonically increasing.

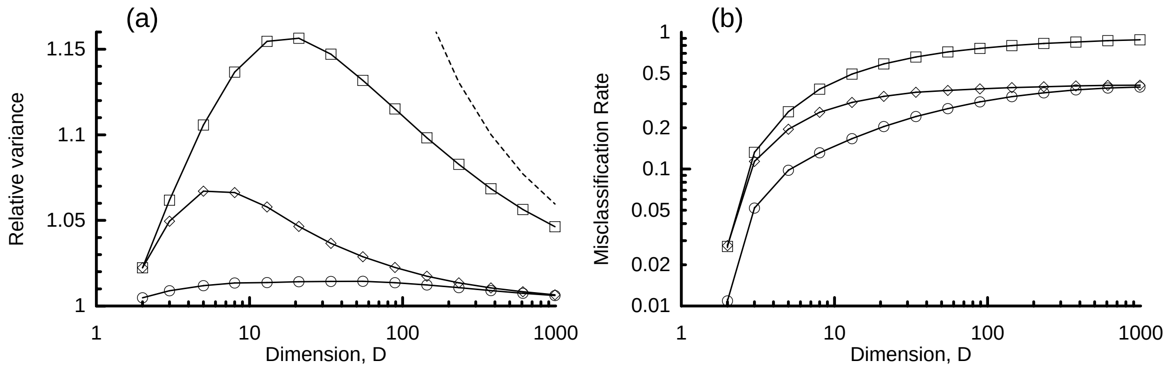

Figure 4. Same as Fig. 3, but for $K = 16$ classes. **(a)** Relative variance plotted against the number D of spectral channels; and **(b)** misclassification rate.

occur when points are nearly equidistant between two candidate centers. One imagines that the process of clustering nonuniform data preferentially produces situations where there are gaps between the clusters, which might reduce the actual rate of misclassifications. Also, although hyperspectral data has nominally a large dimension D, in practice, most of the variance is concentrated in just a few of these dimensions, which effectively reduces the dimensionality of the data. On the other hand, the alternative distance metrics are not rotationally invariant, so it is not clear that they would be able to exploit this lower effective dimensionality.

To compare these metrics in a more realistic situation, we produced a k-means clustering algorithm for which the distance metric was adjustable, and applied it to a set of five AVIRIS data sets. Each data set is a 614×512 image with 224 channels, spanning the range from 0.4 to 2.5 microns. We also used these AVIRIS data sets to produce associated 10 channel data sets simulating the data produced by the MTI sensor.[5]

We initialized the clustering by dividing the image into K equal width "stripes" so that the first N/K points comprise the first cluster, the next N/K points comprise the second cluster, *etc.* We also considered several other initialization strategies, but this one has the advantage that it is simple and deterministic.

Each image was classified with each of the distance measures, first into $K = 2$ clusters, and then into $K = 16$ clusters. Results are shown in Table 1 and Table 2, with discussion in the table captions.

6. CONCLUSIONS

Using both artificial models and real data, we compared the use of the standard Euclidean distance to several cheaper alternative distance metrics for k-means clustering. Although the Euclidean metric is theoretically optimal in terms of the cluster quality, it is also expensive to compute, particularly in a hardware implementation. We considered three alternative distance metrics (Manhattan, Max, and a linear combination of the two) which would be more amenable to hardware implementation. Both in the data-independent Monte-Carlo studies, and in the comparisons using hyperspectral and multispectral image data, we found that the alternative distances generally produced lower quality clusters than those based on Euclidean distance. For the AVIRIS and simulated MTI data, no consistent preference for Manhattan over Max could be concluded, but the Manhattan did have a lower rate of misclassification, and a smaller relative variance increase, in the idealized Monte-Carlo experiments. The linear combination proved better than the Manhattan and Max metrics in the idealized experiments; in practical cases, it was also generally (but not exclusively) better than the other alternatives.

ACKNOWLEDGMENTS

We are grateful to the MTI team for providing both the AVIRIS and the simulated MTI data. We also thank the other members of the ASAPP team for many useful discussions.

REFERENCES

1. NASA, 1999. `http://makalu.jpl.nasa.gov/avaris.html`.
2. V. Castelli and T. M. Cover, "On the exponential value of labeled samples," *Pattern Recognition Lett.* **16**, pp. 105–111, 1995.
3. P.-F. Hsieh and D. Landgrebe, "Statistics enhancement in hyperspectral data analysis using spectral-spatial labeling, the EM algorithm, and the leave-one-out covariance estimator," *Proc. SPIE* **3438**, pp. 183–190, 1999.
4. R. A. Schowengerdt, *Techniques for Image Processing and Classification in Remote Sensing*, Academic Press, Orlando, 1983.
5. P. G. Weber, B. C. Brock, A. J. Garrett, B. W. Smith, C. C. Borel, W. B. Clodius, S. C. Bender, R. R. Kay, and M. L. Decker, "MTI Mission Overview," *Proc. SPIE* **3753**, pp. 340–346, 1999.
6. A. E. Filip, "Linear approximation to $\sqrt{x^2 + y^2}$ having equiribble error characteristics," *IEEE Trans. Audio and Electroacoustics* **21**, pp. 554–556, 1973.
7. A. E. Filip, "A baker's dozen magnitude approximations and their detection statistics," *IEEE Trans. Aerospace and Electronic Systems* **12**, pp. 86–89, 1976.

Image	Within-class Variance				Fractional Difference from Euclidean		
	Euclidean	Manhattan	Max	$\alpha = 0.25$	Manhattan	Max	$\alpha = 0.25$
f960323t01p02_r04_sc01.c.img	0.3227	0.3241	0.3316	0.3239	0.0112	0.0313	0.0104
f970410t01p02_r02_sc02.c.img	0.3959	0.4001	0.3978	0.3996	0.0054	0.0043	0.0049
f970620t01p02_r03_sc02.c.img	0.5265	0.5324	0.5443	0.5319	0.0181	0.0284	0.0170
f970701t01p02_r07_sc01.c.img	0.7334	0.7970	0.7588	0.7959	0.5867	0.1100	0.5905
f970801t01p02_r01_sc01.c.img	0.6331	0.6402	0.6442	0.6386	0.0419	0.0375	0.0350
f960323t01p02_r04_sc01.c.img.mti	0.3148	0.3169	0.3155	0.3152	0.0212	0.0067	0.0096
f970410t01p02_r02_sc02.c.img.mti	0.4087	0.4106	0.4100	0.4088	0.0044	0.0039	0.0010
f970620t01p02_r03_sc02.c.img.mti	0.5344	0.5366	0.5421	0.5347	0.0145	0.0208	0.0045
f970701t01p02_r07_sc01.c.img.mti	0.7178	0.7675	0.7425	0.6999	0.0713	0.6870	0.1377
f970801t01p02_r01_sc01.c.img.mti	0.6497	0.6698	0.7307	0.6593	0.0454	0.2696	0.0232

Table 1. Comparisons of within-class variance for clusterings of 224-channel AVIRIS data (top five lines) and 10-channel simulated MTI data (bottom five lines), using different distance metrics. These variance measurements are divided by the total variance in the data set. $K = 2$ clusters were used, and the algorithm was run for 10 iterations, or until convergence was achieved. In almost all cases, the Euclidean clustering produced the minimum within-class variance (the single exception was the $\alpha = 0.25$ clustering for data set `f970701t01p02_r07_sc01.c.img.mti`). In all cases, the $\alpha = 0.25$ clustering produced smaller variance than the Manhattan metric, though in most cases the difference was small. Although the Manhattan metric is uniformly superior to the Max metric in the idealized Monte-Carlo studies, the comparison for multiple iterations on real data leads to mixed results; the Max was actually better in two of the five AVIRIS images. The difference from the Euclidean clustering is the fraction of pixels which are labeled differently in the Euclidean clustering and in the clustering produced by the alternative distance. This provides an informal measure of "misclassification" but one should be cautious about interpreting that as a measure of goodness. The comparison is with the clustering produced by the Euclidean distance, not with any "ground truth." Even though the algorithm is deterministic and even though each run started with the same initial conditions, small perturbations caused by the use of a different distance metric, can lead after multiple iterations to quite different clusterings.

Image	Within-class Variance			
	Euclidean	Manhattan	Max	$\alpha = 0.25$
f960323t01p02_r04_sc01.c.img	0.0481	0.0738	0.0523	0.0733
f970410t01p02_r02_sc02.c.img	0.0883	0.0878	0.1030	0.0870
f970620t01p02_r03_sc02.c.img	0.0997	0.1154	0.1141	0.1120
f970701t01p02_r07_sc01.c.img	0.1263	0.1325	0.1725	0.1315
f970801t01p02_r01_sc01.c.img	0.0963	0.0958	0.1047	0.0947
f960323t01p02_r04_sc01.c.img.mti	0.0411	0.0424	0.0427	0.0406
f970410t01p02_r02_sc02.c.img.mti	0.0808	0.0826	0.0798	0.0806
f970620t01p02_r03_sc02.c.img.mti	0.0862	0.0957	0.0947	0.0864
f970701t01p02_r07_sc01.c.img.mti	0.1042	0.1001	0.1032	0.1066
f970801t01p02_r01_sc01.c.img.mti	0.0792	0.0789	0.0884	0.0792

Table 2. Comparisons of within-class variance for clusterings of 224-channel AVIRIS data (top five lines) and 10-channel simulated MTI data (bottom five lines), using different distance metrics. These variance measurements are divided by the total variance in the data set. $K = 16$ clusters were used, and the algorithm was run for 50 iterations, or until convergence was achieved.