# Unsupervised Clustering of Hyperspectral Images Using Diffusion Maps with Mahalanobis Distances

Edward Li and James M. Murphy

***Abstract*—Computing a diffusion map first requires calculating local similarity measures between data points. In these similarity measures, a choice of distance metrics must be made. Hyperspectral images consist of individual measurements across a multitude of spectral bands, and the distributions within each spectral can vary based on the nature of the materials they represent. A Euclidean distance metric may not account for this and treat variations in each spectral with the same significance. Thus, it would be beneficial to use a distance metric that normalizes the variances across all spectral bands. In this paper, we explore how the Mahalanobis distance metric may achieve this goal.**

## I. Introduction

THE problem of hyperspectral clustering requires an algorithm to label each pixel in a hyperspectral image as belonging to a class without access to any labeled data. This can be very challenging unless further assumptions are made [1]. The methods proposed in this paper assume that the hyperspectral images contain classes of data points which are normally distributed with independent, anisotropic covariance matrices. We exploit the characteristics of the Mahalanobis distance metric in constructing the connectivity matrices for our diffusion processes.

## II. Unsupervised Learning Algorithm

### A. Motivations

Diffusion maps are used because they have experimentally shown significant improvements over other state-of-the-art methods for hyperspectral clustering. They also have robust theoretical performance guarantees [2].

Each spectral band of a hyperspectral image has its own associated variance that may be different from that of another spectral band. In addition, multiple spectral bands may be linked to the same class, causing them to vary together. Thus, Mahalanobis distances are used to normalize the distances in the original data set based on its covariances.

Estimating covariances on the overall dataset can be less meaningful due to the existence of multiple independent distributions, each associated with a different class. Thus, we need a method of estimate covariances on a local basis. Like most real-world data sets, the classes in a hyperspectral image follow Gaussian distributions [3]. Assuming that each class of data points has an underlying Gaussian distribution, we can use Gaussian mixture models to determine where each covariance matrix should be estimated locally. Furthermore, Gaussian mixture models by themselves have been used for hyperspectral clustering with promising results [4].

Advisor: James M. Murphy, Department of Mathematics, The Johns Hopkins University.

### B. Diffusion Maps

Consider a discrete set of $N$ elements $X \subset \mathbb{R}^D$. Diffusion maps reorganize data based on the underlying geometry. By combining heat kernels with a random walk, diffusion maps achieve nonlinear dimensionality reduction [5].

First, the connectivity between pairs of data points is determined using a local similarity measure. The connectivity between two data points, $X_i, X_j \in X$, is represented by the probability of moving between $X_i$ and $X_j$ in one step of the random walk:

$$\text{connectivity}(X_i, X_j) = P(X_i, X_j). \tag{1}$$

To calculate the probability function, we need to define a local similarity measure within the neighborhood of each point. We begin with a non-normalized likelihood function, $k$, which we will call the diffusion kernel. In addition to quickly going to zero outside of the neighborhood of each point, the diffusion kernel needs to satisfy the following properties:

1) $k$ is positivity preserving: $k(x, y) \geq 0$,
2) $k$ is symmetric: $k(x, y) = k(y, x)$.

The first property allows us to normalize the kernel matrix (whose entries are just the diffusion kernels between each pair of data points) into a probability matrix. The second property allows us to perform spectral analysis of the probability matrix [6].

In our proposed method, we use the Gaussian kernel,

$$k(X_i, X_j) = \exp\left(-\frac{d(X_i, X_j)^2}{\sigma}\right) \tag{2}$$

where $d(X_i, X_j)$ is the choice of distance metric. Here, we can define the neighborhood of $X_j$ as all $X_j$ for which $k(X_i, X_j) \geq \epsilon$ where $0 < \epsilon \ll 1$. The size of the neighborhood is adjusted by tweaking $\sigma$. We want the neighborhood to be the area in which we can trust our distance metric to capture the local geometry accurately, which depends on the density of the data.

Using our choice of kernel, we can construct a $N \times N$ weight matrix $W$, which is defined as

$$W_{ij} = \begin{cases} k(X_i, X_j), & X_j \in NN_k(X_i) \\ 0, & else \end{cases} \tag{3}$$

where $NN_k(X_i)$ is the set of k-nearest neighbors of $X_i$ with respect to Euclidean distance. To get to the Markov probability matrix for the diffusion process, we normalize $W$ to be row-stochastic:

$$P_{ij} = \frac{W_{ij}}{\sum_k W_{ij}}. \tag{4}$$

For the diffusion process, $P$ gives the probabilities for a single step taken from $X_i$ to $X_j$. To increase the number of steps taken, we can simply take powers of $P$. Thus, $P^t_{ij}$ gives the sum of the probabilities of all paths of length $t$ from $X_i$ to $X_j$.

Next, we want to define the diffusion distance, which is a metric on the underlying structure revealed by the diffusion process. This metric is related to $P$ and is given by

$$D_t(X_i, X_j) = \sqrt{\sum_k |P^t_{ik} - P^t_{jk}|^2}. \tag{5}$$

The term $|P^t_{ik} - P^t_{kj}|^2$ is minimized when there are similar probability paths of length $t$ to $X_k$ from both $X_i$ and $X_j$. This occurs when $X_i$ and $X_j$ are well-connected to begin with, i.e. there is a large number of shorter, high-probability paths connecting $X_i$ and $X_j$. Thus, the diffusion distance is small when $X_i$ and $X_j$ are in close proximity along the underlying geometry. Since the diffusion distance sums over all possible paths of length $t$ connecting $X_i$ and $X_j$ to any point in the data set, it takes into account all evidences relating $X_i$ and $X_j$ and is robust to noise pertubation [5].

To reorganize the data according to the diffusion distance metric, we map coordinates between the data and the diffusion space:

$$Y_i \coloneqq \begin{bmatrix} P^t_{i1} \\ P^t_{i2} \\ \vdots \\ P^t_{iN} \end{bmatrix}. \tag{6}$$

Taking the Euclidean distance between two mapped points $Y_i$ and $Y_j$, we recover the diffusion distance between $X_i$ and $X_j$:

$$\begin{aligned} \sqrt{|Y_i - Y_j|^2} &= \sqrt{\sum_k |P^t_{ik} - P^t_{jk}|^2} \\ &= D_t(X_i, X_j). \end{aligned}$$

Note that the dimensionality of the mapped data is the sample size, $N$. By ignoring certain dimensions in the diffusion space, we can reduce the dimensionality of the data. To determine which dimensions are best for ignoring, we first express (6) in terms of the eigenvectors and eigenvalues of $P$:

$$Y_i = \begin{bmatrix} \lambda^t_1 \Phi_1(i) \\ \lambda^t_2 \Phi_2(i) \\ \vdots \\ \lambda^t_N \Phi_N(i) \end{bmatrix} \tag{7}$$

where $\Phi_k(i)$ indicates the $i$-th element of the $k$-th eigenvector of $P$. The eigenvectors form a basis for the diffusion space and their corresponding eigenvalues indicate the importance of each dimension. Since the eigenvalues are organized in decreasing order from top to bottom, we can simply take the top $m < N$ eigenvectors to approximate the diffusion distance. Thus, we have the dimensionality-reduced diffusion mapping

$$Y'_i = \begin{bmatrix} \lambda^t_1 \Phi_1(i) \\ \lambda^t_2 \Phi_2(i) \\ \vdots \\ \lambda^t_m \Phi_m(i) \end{bmatrix}, \tag{8}$$

which optimally preserves the intrinsic geometry of the data [6].

### C. Mahalanobis Distances

Consider a point $x \in \mathbb{R}^D$ and a distribution $X \subset \mathbb{R}^D$. The Mahalanobis distance is a multidimensional generalization of measuring how many standard deviations away $x$ is from the mean $\mu$ of $X$. With the covariance matrix $\Sigma$ of the distribution $X$, the Mahalanobis distance is defined as

$$d_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}. \tag{9}$$

in [7]. This metric is zero if $x$ is at $\mu$ and grows as $x$ moves away from $\mu$ along each principal component axis. If $\Sigma$ is the identity matrix, then the Mahalanobis distance reduces to the Euclidean distance:

$$\begin{aligned} \sqrt{(x-\mu)^T I_D^{-1} (x-\mu)} &= \sqrt{(x-\mu)^T (x-\mu)} \\ &= d_E(x). \end{aligned}$$

For a $D$-dimensional multivariate distribution in Euclidean space, the closer a point is to its center of mass, the more likely it is to belong to the distribution. A simplistic approach to measuring how likely a given point is to the distribution involves estimating the standard deviation of the distances of sample points from the center of mass. However, this approach assumes that the sample points are distributed about the center of mass in a spherical manner and does not take into account the difference in covariances along different dimensions. On the other hand, the Mahalanobis distance does not assume a spherical distribution, so the probability of a point belonging to the distribution depends on both the distance and the direction from the center of mass. Thus, using the Mahalanobis distance, a difference in Euclidean distance along an axis of low variance will mean a lot more than a difference in Euclidean distance along an axis of high variance.

### D. Gaussian Mixture Models

Gaussian mixture models are a form of unsupervised learning which can be used to represent normally distributed subpopulations within an overall population. For a Gaussian mixture model with $K$ components, the parameters are the mixture component weights $\phi_{1,\dots,K}$, where $\sum_{i=1}^K \phi_i = 1$, and the component means $\mu_{1,\dots,K}$ and covariance matrices $\Sigma_{1,\dots,K}$. These parameters are learned on the data using the expectation maximization technique, after which the component probabilities are given by

$$p(x) = \sum_{i=1}^K \phi_i \mathcal{N}(x \mid \mu_i, \Sigma_i). \tag{10}$$

in [8].

Fitting multimodal distributions with a unimodal model generally produces poor results. For a better fit, mixture models assume that the multimodal distribution is generated by multiple unimodal distributions. Additionally, real-world unimodal data is usually best modeled using Gaussian distributions for many theoretical reasons. Thus, representing multimodal distributions using a mixture of many unimodal Gaussian distributions makes intuitive sense.

### E. Proposed Algorithm

Our diffusion mapping algorithm first constructs a k-d tree to find the $k$-nearest neighbors for all $N$ data points, taken as each $d$-dimensional pixel of the hyperspectral image. Then, it computes a Gaussian kernel matrix between every pair of data points, where the distance metric used in the kernel can be either Euclidean or Mahalanobis. After normalizing each row of the kernel matrix to get a Markov probability matrix, the $m$ most dominant eigenvectors are calculated. To get the dimensionality-reduced diffusion mapping, each eigenvector is multiplied by its corresponding eigenvalue taken to the $t$-th power, where $t$ is the number of time steps taken. Finally, we use $K$-means clustering on the diffusion mapped points to construct our resulting classes.

Algorithm 1 is the classic diffusion mapping using Euclidean distances in its local similarity measures.

**Algorithm 1:** Diffusion Map Using Euclidean Distance

**Input :** High-dimensional data set $X \in \mathbb{R}^D$ with $N$ data points
**Output:** Low-dimensional data set $Y' \in \mathbb{R}^m$ with $N$ data points

1 Construct a $N \times N$ kernel matrix $W$ such that $k(i,j) = \exp\left(-\frac{d_E(X_i,X_j)^2}{\sigma}\right)$.
2 Normalize each row of $W$ to get the Markov probability matrix for the diffusion process.
3 Calculate the top $m$ eigenvectors of the Markov probability matrix.
4 Multiply each eigenvector by the $t$-th power of its corresponding eigenvalue to get a mapping to the diffusion space at time step $t$.

Unlike the Euclidean-based diffusion mapping, where the kernel matrix can be calculated directly from the Euclidean distances in the original data set, the Mahalanobis-based diffusion mappings require a covariance matrix to be estimated before calculating the Mahalanobis distances to be used in the kernel matrix. We present two approaches to doing this: one where the covariance matrix is calculated on the overall data and another where multiple covariance matrices are calculated on the clusters returned by fitting a Gaussian mixture model to the overall data.

Algorithm 2 requires a $D \times D$ covariance matrix to be estimated on the overall data set and then inverted. Due to the $O(D^3)$ time complexity of Gaussian-Jordan elimination used for matrix inversions [9], this does not scale well with higher-dimensional data sets. Additionally, calculating just one covariance matrix on the overall data set may be meaningless due to the existence of separate classes with different means.

Algorithm 3 requires a separate $D \times D$ covariance matrix to be estimated and inverted for each Gaussian mixture model cluster. Due to the $O(D^3)$ time complexity of Gaussian-Jordan elimination used for matrix inversions, this scales even worse with higher-dimensional data sets than 2 does. However, calculating a separate covariance matrix for each class makes more sense because each class may have difference covariances.

**Algorithm 2:** Diffusion Map Using Mahalanobis Distance with Overall Covariance Matrix

**Input :** High-dimensional data set $X \in \mathbb{R}^D$ with $N$ data points
**Output:** Low-dimensional data set $Y' \in \mathbb{R}^m$ with $N$ data points

1 Estimate the covariance matrix $\Sigma$ of $X$.
2 Construct a $N \times N$ kernel matrix $W$ such that $k(i,j) = \exp\left(-\frac{d_M(X_i,X_j)^2}{\sigma}\right)$, where $d_M$ uses $\Sigma$ as its covariance matrix.
3 Normalize each row of $W$ to get the Markov probability matrix for the diffusion process.
4 Calculate the top $m$ eigenvectors of the Markov probability matrix.
5 Multiply each eigenvector by the $t$-th power of its corresponding eigenvalue to get a mapping to the diffusion space at time step $t$.

**Algorithm 3:** Diffusion Map Using Mahalanobis Distance with Local Covariance Matrices

**Input :** High-dimensional data set $X \in \mathbb{R}^D$ with $N$ data points
**Output:** Low-dimensional data set $Y' \in \mathbb{R}^m$ with $N$ data points

1 Fit a Gaussian mixture model to $X$ with $c$ clusters.
2 For each of the $c$ clusters, estimate the covariance matrix $\Sigma_k$.
3 Construct a $N \times N$ kernel matrix $W$ such that $k(i,j) = \exp\left(-\frac{d_{M_{ij}}(X_i,X_j)^2}{\sigma}\right)$, where $d_{M_{ij}}$ uses $\Sigma_{k_i} + \Sigma_{k_j}$ as its covariance matrix and $k_i$ corresponds to the cluster that contains $X_i$.
4 Normalize each row of $W$ to get the Markov probability matrix for the diffusion process.
5 Calculate the top $m$ eigenvectors of the Markov probability matrix.
6 Multiply each eigenvector by the $t$-th power of its corresponding eigenvalue to get a mapping to the diffusion space at time step $t$.

Because we rely on a Euclidean-based nearest neighbor search to determine which pairs of data points are included in the weight matrix $W$, Algorithms 2 and 3 require a greater number of nearest neighbors to ensure that the connectivity of the weight matrix is not entirely based on the proximity of the points in Euclidean space. In future implementations of Algorithms 2 and 3, we will use a nearest neighbor search based on the Mahalanobis distance instead if it proves to be more efficient overall. However, for the purposes of our investigation, using a Euclidean-based nearest neighbor search with a high enough number of neighbors is sufficient.

## III. Experiments

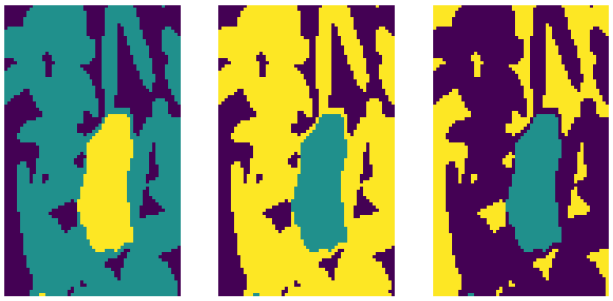
### A. Rice Data

The rice data set is a 56 by 93 pixel image consisting of 101 spectral bands. There are 3 groundtruth classes: background, rice, and plastic contaminant. We use the parameters in Table I for the Euclidean- and Mahalanobis-based diffusion maps.

TABLE I: Rice Diffusion Map Parameters

| | Euclidean | Global Mahalanobis | Local Mahalanobis |
|---|---|---|---|
| Kernel tweak | $\sigma = 1e3$ | $\sigma = 1e6$ | $\sigma = 1e6$ |
| Neighbors | $k = 500$ | $k = 500$ | $k = 500$ |
| Eigenvectors | $m = 3$ | $m = 3$ | $m = 3$ |
| Time steps | $t = 5$ | $t = 5$ | $t = 5$ |
| GMM clusters | N/A | N/A | $c = 10$ |

Fig. 1: Luminance map of the rice data.



(a) Euclidean (b) Global mahalanobis (c) Local mahalanobis

Fig. 2: Clustering of the rice data.

### B. Indian Pines Data

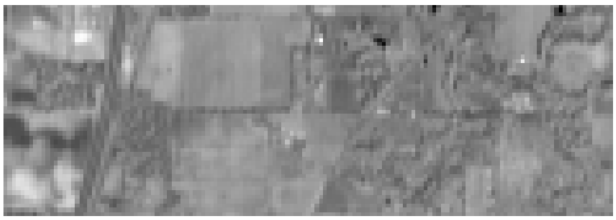
We took a subset of the Indian Pines data set that is a 145 by 50 pixel image consisting of 200 spectral bands. There are 8 groundtruth classes. We use the parameters in Table II for the Euclidean- and Mahalanobis-based diffusion maps.

### C. Salinas-A Data

The Salinas-A data set is a subset of the Salinas data set. It is a 83 by 86 pixel image consisting of 204 spectral bands. There are 7 groundtruth classes. We use the parameters in Table III for the Euclidean- and Mahalanobis-based diffusion maps.

TABLE II: Indian Pines Diffusion Map Parameters

| | Euclidean | Global Mahalanobis | Local Mahalanobis |
|---|---|---|---|
| Kernel tweak | $\sigma = 1e7$ | $\sigma = 1e7$ | $\sigma = 1e7$ |
| Neighbors | $k = 500$ | $k = 500$ | $k = 500$ |
| Eigenvectors | $m = 10$ | $m = 10$ | $m = 10$ |
| Time steps | $t = 30$ | $t = 30$ | $t = 30$ |
| GMM clusters | N/A | N/A | $c = 10$ |



(a) Luminance Map

(b) Groundtruth

Fig. 3: The Indian Pines data.

TABLE III: Salinas-A Diffusion Map Parameters

| | Euclidean | Global Mahalanobis | Local Mahalanobis |
|---|---|---|---|
| Kernel tweak | $\sigma = 1e6$ | $\sigma = 1e6$ | $\sigma = 1e7$ |
| Neighbors | $k = 500$ | $k = 500$ | $k = 500$ |
| Eigenvectors | $m = 10$ | $m = 10$ | $m = 10$ |
| Time steps | $t = 30$ | $t = 30$ | $t = 30$ |
| GMM clusters | N/A | N/A | $c = 10$ |

### D. Samson Data

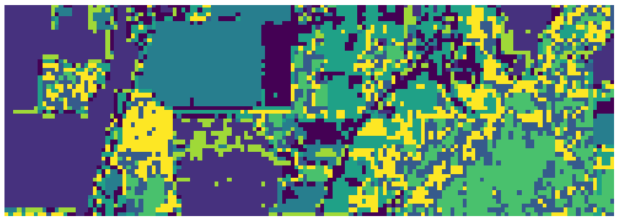
We took a subset of the Samson data set that is a 95 by 95 pixel image consisting of 156 spectral bands. There are 3 groundtruth classes. We use the parameters in Table IV for the Euclidean- and Mahalanobis-based diffusion maps.

TABLE IV: Samson Diffusion Map Parameters

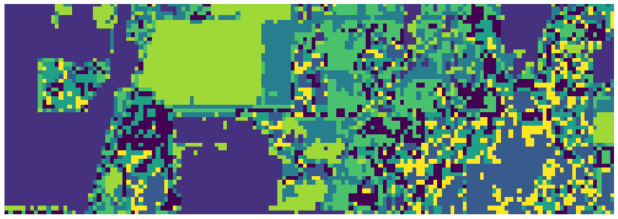
| | Euclidean | Global Mahalanobis | Local Mahalanobis |
|---|---|---|---|
| Kernel tweak | $\sigma = 1e6$ | $\sigma = 1e6$ | $\sigma = 1e9$ |
| Neighbors | $k = 500$ | $k = 500$ | $k = 500$ |
| Eigenvectors | $m = 10$ | $m = 10$ | $m = 10$ |
| Time steps | $t = 30$ | $t = 30$ | $t = 30$ |
| GMM clusters | N/A | N/A | $c = 10$ |

### E. Jasper Ridge Data

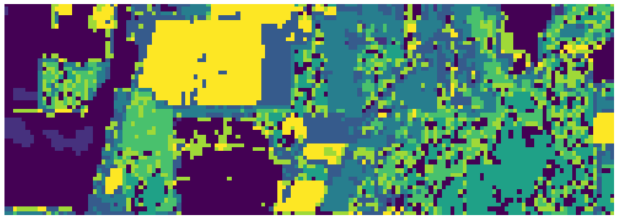
We took a subset of the Jasper Ridge data set that is a 100 by 100 pixel image consisting of 98 spectral bands. There are 4 groundtruth classes. We use the parameters in Table V for the Euclidean- and Mahalanobis-based diffusion maps.



(a) Euclidean

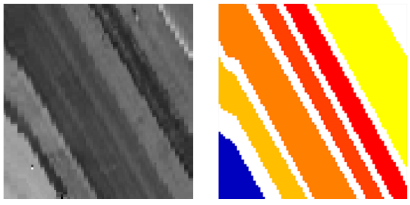


(b) Global mahalanobis



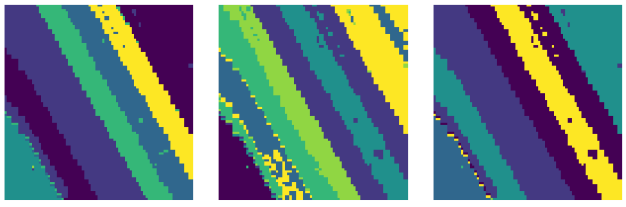
(c) Local mahalanobis

Fig. 4: Clustering of the Indian Pines data.



(a) Luminance map (b) Groundtruth

Fig. 5: The Salinas-A data.

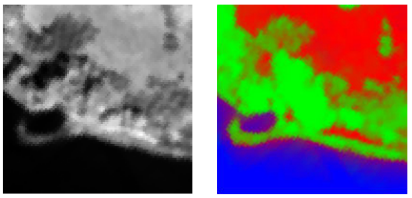


(a) Euclidean (b) Global mahalanobis (c) Local mahalanobis

Fig. 6: Clustering of the Salinas-A data.

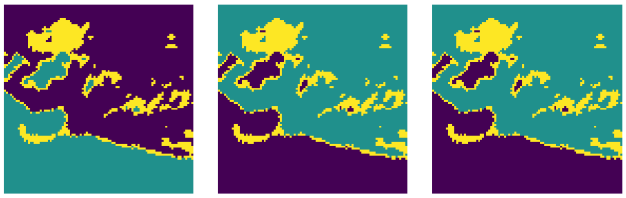
## IV. Conclusion

In our experiments, we have not seen any significant improvements in using the Mahalanbois distance in our local similarity measures over the Euclidean distance. The reason behind this is that the diffusion process only uses Mahalanobis distance in its local similarity measures to capture local features which are not susceptible to the effects of having anisotropic covariance matrices. Whether we calculate the



(a) Luminance map (b) Groundtruth

Fig. 7: The Samson data.

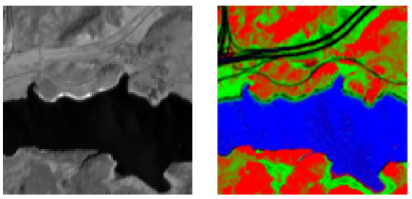


(a) Euclidean (b) Global Mahalanobis (c) Local Mahalanobis

Fig. 8: Clustering of the Samson data.

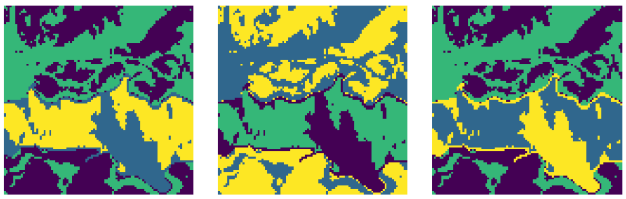
TABLE V: Jasper Ridge Diffusion Map Parameters

| | Euclidean | Global Mahalanobis | Local Mahalanobis |
|---|---|---|---|
| Kernel tweak | $\sigma = 1e9$ | $\sigma = 1e6$ | $\sigma = 1e9$ |
| Neighbors | $k = 2500$ | $k = 2500$ | $k = 2500$ |
| Eigenvectors | $m = 10$ | $m = 10$ | $m = 10$ |
| Time steps | $t = 30$ | $t = 30$ | $t = 30$ |
| GMM clusters | N/A | N/A | $c = 10$ |



(a) Luminance map (b) Groundtruth

Fig. 9: The Jasper Ridge data.



(a) Euclidean (b) Global Mahalanobis (c) Local Mahalanobis

Fig. 10: Clustering of the Jasper Ridge data.

local similarities with the Euclidean or Mahalanobis distance metric, the diffusion process captures the connectivity of the underlying geometry. Any deficiencies with the Euclidean-based diffusion process can be compensated for by adjusting the kernel parameter.

Additionally, the Mahalanobis-based diffusion maps run slightly slow than their Euclidean-based counterparts due to the additional steps that must be taken. Before computing the Mahalanobis distances, we need to calculate and invert covariance matrices, which takes $O(D^3)$ time and severely limits the number of dimensions our data can have.

For the aforementioned reasons, we do not see any valid reason to use anything other than Euclidean distance metric in the local similarity measures of the diffusion process.

## REFERENCES

[1] J.M. Murphy and M. Maggioni. Nonlinear Unsupervised Clustering of Hyperspectral Images with Applications to Anomaly Detection and Active Learning. 2017.

[2] M. Maggioni and J.M. Murphy. Clustering by unsupervised geometric learning of modes. 2017.

[3] D. Manolakis, D. Marden, J. Kerekes, G. Shaw. On the statistics of hyperspectral imaging data. *Proc. SPIE*, 2001.

[4] N. Acito, G. Corsini, and M. Diani. An unsupervised algorithm for hyperspectral image segmentation based on the gaussian mixture model. *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 6, pp. 3745–3747, 2003.

[5] R.R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):530, 2006.

[6] J. de la Porte, B.M. Herbst, W. Hereman, S.J. van der Walt. An Introduction to Diffusion Maps. 2008.

[7] R. De Maesschalck, D. Jouan-Rimbaud, D.L. Massart. The Mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000.

[8] D.A. Reynolds. Gaussian Mixture Models. *Encyclopedia of Biometrics*, 2009.

[9] R.W. Farebrother. *Linear Least Squares Computations*. CRC Press, 1988.