Signal Ensemble Classification using Low-Dimensional Embeddings and Earth Mover's Distance

Linh Lieu and Naoki Saito

Abstract Instead of classifying individual signals, we address classification of objects characterized by signal ensembles (i.e., collections of signals). Such necessity arises frequently in real situations: e.g., classification of video clips or object classification using acoustic scattering experiments to name a few. In particular, we propose an algorithm for classifying signal ensembles by bringing together well-known techniques from various disciplines in a novel way. Our algorithm first performs the dimensionality reduction on training ensembles using either the linear embeddings (e.g., Principal Component Analysis (PCA), Multidimensional Scaling (MDS)) or the nonlinear embeddings (e.g., the Laplacian eigenmap (LE), the diffusion map (DM)). After embedding training ensembles into a lower-dimensional space, our algorithm extends a given test ensemble into the trained embedding space, and then measures the "distance" between the test ensemble and each training ensemble in that space, and classify it using the nearest neighbor method. It turns out that the choice of this ensemble distance measure is critical, and our algorithm adopts the so-called Earth Mover's Distance (EMD), a robust distance measure successfully used in image retrieval and image registration. We will demonstrate the performance of our algorithm using two real examples: classification of underwater objects using multiple sonar waveforms; and classification of video clips of digit-speaking lips. This article also provides a concise review on the several key concepts in statistical learning such as PCA, MDS, LE, DM, and EMD as well as the practical issues including how to tune parameters, which will be useful for the readers interested in numerical experiments.

Linh Lieu
Department of Mathematics, University of California, One Shields Avenue, Davis, CA 95616, e-mail: `llieu@math.ucdavis.edu`

Naoki Saito
Department of Mathematics, University of California, One Shields Avenue, Davis, CA 95616, e-mail: `saito@math.ucdavis.edu`

1 Introduction

Many problems in pattern recognition often require comparison between ensembles of signals (i.e., points in a high-dimensional space) instead of comparing individual signals. For example, in a visual speech recognition problem, two or more clips of recorded video are compared for similar patterns. Typically a video clip consists of a sequence of images called video frames. We can view a video frame as a point and a video clip as an ensemble of points in the high-dimensional image space. Thus, comparing two video clips is equivalent to comparing two ensembles of points in the image space. As a second example, consider the task of identifying an object on the ocean floor given a set of sonar waveforms reflected from the object. This requires comparing the ensembles of waveforms reflected from the unknown object with those reflected from known objects. Then the unknown object is identified when a match is made.

In this article, we propose an algorithm for classifying such signal ensembles. More specifically, let $\mathscr{X} := \bigcup_{i=1}^{M} \mathscr{X}^i \subset \mathbb{R}^d$ be a collection of M training *ensembles* $\mathscr{X}^i := \{\mathbf{x}_1^i, \cdots, \mathbf{x}_{m_i}^i\}$, $i = 1, \cdots, M$ where $m_i \in \mathbb{N}$ is the number of signals contained in the ensemble $\mathscr{X}^i$ and $\mathbf{x}_k^i \in \mathbb{R}^d$ is an individual signal (a vector of length d). We assume that each training ensemble $\mathscr{X}^i$ has a unique label (or class name) among C possible labels. Let $\mathscr{Y} := \bigcup_{j=1}^{N} \mathscr{Y}^j \subset \mathbb{R}^d$ be a collection of test (i.e., unlabeled) ensembles where $\mathscr{Y}^j := \{\mathbf{y}_1^j, \cdots, \mathbf{y}_{n_j}^j\}$ is the jth test ensemble consisting of n_j signals of length d. Our goal is to classify each $\mathscr{Y}^j$, $j = 1, \cdots, N$, to one of the possible C classes given the training ensembles $\mathscr{X}$. Note that this task is different from classifying each signal $\mathbf{y}_k^j \in \mathscr{Y}$ individually.

Our proposed algorithm consists of two main stages. The first (or training) stage performs the dimensionality reduction without losing important features of the training ensembles followed by constructing a compact representation called a *signature* — a generalized version of a histogram — that well represents an essence of the ensemble in the reduced space. The second (or classification) stage embeds a given test ensemble into the reduced space obtained in the first stage followed by classifying it to the label of the training ensemble whose signature is most similar to that of the test ensemble. Here how to define the similarity or the distance measure between signatures will be the key issue as we will discuss below.

1.1 Dimensionality reduction

Let us now discuss the first stage in more details. The dimensionality reduction of the input data is necessary because modern technologies often generate data of extremely high dimension. For example, a small 128×128 gray-scale image has dimension $d = 16384$. Such high dimensionality makes data analysis inefficient and inaccurate. Fortunately, the data that we encounter often have low intrinsic dimensionality. For example, consider the set of all 128×128 gray-scale images taken of

an object under fixed lighting by a moving camera. This is a subset of $\mathbb{R}^{16384}$ possessing the natural structure of a low-dimensional manifold with its dimensionality defined by the degrees of freedom of the camera [3]. In short, it is desirable and in fact indispensable to find an effective low-dimensional representation of the data for the task at hand.

Perhaps, the most well-known techniques are Principal Component Analysis (PCA) also known as Karhunen-Loève Transform (KLT) [19, Chap. 8], [17, Sec. 14.5], and Multidimensional Scaling (MDS) [7], [19, Chap. 14], [17, Sec. 14.8]. Despite their shortcomings (e.g., inability to capture *local* geometric information), they have been the mainstay of dimensionality reduction. They have been applied to a variety of applications including human face characterization [18, 35], pose estimation of 3D objects from image sequences [28], to name only a few; see also [17, Sec. 14.5] for more examples.

Many nonlinear methods for dimensionality reduction have been proposed in order to improve the shortcomings of PCA/MDS; see e.g., [32, 12, 41] for some specific proposals and [25] for a comparative review on many such techniques. Unlike classical methods such as PCA and MDS, nonlinear methods in general offer the advantage of preserving local geometry while achieving dimensionality reduction. Providing the example of a moving camera in the paragraph above for motivation, Belkin and Niyogi [2, 3, 4] were among the group of scientists who proposed a nonlinear dimensionality reduction algorithm that explicitly considers the manifold structure that may very well be the intrinsic geometry of the data. They proposed the *Laplacian eigenmap* (LE): an embedding of data into a reduced space spanned by the eigenvectors of the graph Laplacian defined on a similarity graph constructed from the training data while preserving the local geometry of the data. If data arise from non-uniform sampling of a manifold, however, embedding via a Laplacian eigenmap may result in distortion of the manifold when embedded into the reduced space (see [22] for some examples). Sensitivity to sampling densities may be a serious drawback in certain cases. For this reason, Coifman and Lafon [9, 22] proposed a density-invariant normalization of the weights on the graph before computing the graph Laplacian. This would eliminate sensitivity to sampling densities of the Laplacian eigenmaps. Furthermore, the authors defined *diffusion maps* from the eigenvalues and eigenvectors of the diffusion operator (defined in Eq. (14) below) and provided an intuitive interpretation of how point clustering in a diffusion coordinate system is linked to a Markov random walk on the weighted graph; see e.g., [21] for the details.

In this article, we will particularly focus on PCA, MDS, Laplacian eigenmaps, and diffusion maps as our dimensionality reduction methods and compare their performance.

1.2 Nearest neighbor classification and its base distance measure

The final step of the second stage of our proposed algorithm is to actually classify each test ensemble into one of the possible classes by comparing its signature with those of the training ensembles computed in the first stage. The label of the training ensemble whose signature is "closest" to that of a given test ensemble is assigned to it. In other words, our algorithm adopts the so-called k-nearest neighbor (k-NN) classifier with $k = 1$ on the signatures in the reduced embedding space. The k-NN classifier has been quite successful in many classification problems despite its simplicity; see e.g., [17, Sec. 13.3] for the details. Although our algorithm can easily accommodate a different value of $k > 1$, we will stick with $k = 1$ in this article mainly because: 1) choosing the optimal k requires a relatively large number of training ensembles; and 2) a famous result of Cover and Hart [11] claims that the error rate of the 1-NN classifier is never more than twice the Bayes (i.e., optimal) rate asymptotically.

The more important issue in this second stage is the choice of the base similarity or distance measure for the nearest neighbor classifier. There are many such measures for comparing given two signatures: e.g., the usual Euclidean distance, the Hellinger distance, the Kullback-Leibler divergence, etc.; see [1] for the review of many of these distances. The most robust one that is also most suitable for our problem turns out to be the so-called *Earth Mover's Distance* (EMD), which has been successfully applied to many practical problems including image retrieval from a large image database [33, 34], image registration and warping [16], to name a few. To the best of our knowledge, however, using EMD for classification of objects characterized by signal ensembles has not been addressed by others. In our case, each signature may consists of different number of clusters or "bins" in the reduced embedding space. Hence, the conventional distance measures such as the Euclidean distance, the Hellinger distance, etc., are not suitable for our classification problem.

1.3 Related Works

In this subsection, we will briefly describe two works most related to ours. In [42], the authors presented a method to characterize signal ensembles using various probabilistic distance measures in the reproducing kernel Hilbert space (RKHS). First, the signal ensembles are mapped to an RKHS via a nonlinear map using either a polynomial kernel or the Gaussian kernel. Then the probabilistic distance measures (e.g., the Bhattacharyya distance, the Kullback-Leibler divergence, the Chernoff distance, etc.) are computed in the RKHS space to determine the similarities between signal ensembles. Our proposed method and [42] share the common objective of classifying signal ensembles. Other than using a nonlinear mapping to transform the data from its original space, however, our algorithm and [42] are two completely different approaches. For example, the approach of [42] assumes that the data after mapped to an RKHS obey the normal (or Gaussian) distribution, and they proceed to

estimate the parameters (the mean vector and the covariance matrix) of the normal distribution. Such an assumption is too strong and cannot be guaranteed in practice in our opinion even if one finds a good nonlinear kernel function.

In [20], Lafon, Keller, and Coifman utilized diffusion maps and the out-of-sample extension scheme for signal ensemble matching problems. Our proposed algorithm owes them quite a bit: we too use diffusion maps and their out-of-sample extension scheme, which will be reviewed in Sec. 2.4 in detail. An important difference between our algorithm and theirs, however, is the use of EMD as the similarity/distance measure instead of the Hausdorff distance (HD) they adopted; see Eq. (22) for the definition of HD and Sec. 3 for the detailed discussion. We also note that the techniques we employ in the signature construction stage for EMD-based classification is not limited to the framework of the diffusion maps; in fact, we also use the Laplacian eigenmaps and PCA in our numerical experiments below. Furthermore, the advantage of EMD over HD is its robustness to outliers. As we will see in Sec. 5 below, HD is very sensitive to outliers. In contrast, the EMD between the signatures measures the differences between the distributions of points within the reduced space. Therefore it is less affected by outliers, which leads to more reliable results.

1.4 Article Organization

In Section 2, we will review the basics of the particular low-dimensional embedding techniques for dimensionality reduction, i.e., Multidimensional Scaling/Principal Component Analysis, the Laplacian eigenmaps, and the diffusion maps. Section 3 reviews the key player in our proposed algorithm: the Earth Mover's Distance. Section 4 describes our proposed algorithm for signal ensemble classification in detail including the practical issues such as the tuning parameter selection that are important to successfully embed high-dimensional signals into the lower dimensional spaces using the diffusion maps/Laplacian eigenmaps. Section 5 describes the results of our numerical experiments on two real datasets: underwater object classification using acoustic signals and classification of video clips of digit-speaking lips. Finally, we will conclude in Section 6 with discussion.

2 Dimensionality Reduction/Low-Dimensional Embeddings

In this section, we will review the embeddings of high-dimensional data into a low-dimensional space using MDS/PCA, Laplacian eigenmaps, and diffusion maps and discuss their properties that allow us to achieve meaningful dimensionality reduction as well as their similarity and difference. We will also review an algorithm proposed in [20] for extending these embedding maps from training data to test data.

First of all, let $\mathscr{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_{m_\star}\} \subset \mathbb{R}^d$ be a set of the available training data points (or signals). Note that using the notations defined in Introduction, the total number of training data points $m_\star := \sum_{i=1}^{M} m_i$. In this article, we assume the finite number of data points both for training and test signals. Since $\mathscr{X} \subset \mathbb{R}^d$, we can define a natural dissimilarity measure δ that gives a sense of affinity between any two data points in $\mathscr{X}$. The usual Euclidean distance is most commonly used as δ although there are many other possibilities. Let X be a data matrix of size $d \times m_\star$ (i.e., each column vector is a training signal)[1], and let $\Psi : \mathscr{X} \subset \mathbb{R}^{m_\star} \to \mathbb{R}^s$ be an embedding map from high to low dimensional spaces where $1 \leq s \ll \min(d, m_\star)$ is the reduced dimension. With a slight abuse of notation, let $\Psi(X)$ denote a matrix of embedded points of $\mathscr{X}$ in $\mathbb{R}^s$, i.e.,

$$\Psi(X) := (\Psi(\mathbf{x}_1), \ldots, \Psi(\mathbf{x}_{m_\star})) = \begin{bmatrix} \psi_1^T \\ \vdots \\ \psi_s^T \end{bmatrix} = \begin{bmatrix} \psi_1(\mathbf{x}_1) & \ldots & \psi_1(\mathbf{x}_{m_\star}) \\ \vdots & & \vdots \\ \psi_s(\mathbf{x}_1) & \ldots & \psi_s(\mathbf{x}_{m_\star}) \end{bmatrix} \in \mathbb{R}^{s \times m_\star}. \quad (1)$$

2.1 Multidimensional Scaling and Principal Component Analysis

Multidimensional Scaling (MDS) [7], [19, Chap. 14], [17, Sec. 14.8] seeks a low-dimensional embedding that minimizes the following quantity:

$$J_{\text{MDS}}(\Psi) := \sum_{i=1}^{m_\star} \sum_{j=i}^{m_\star} \left(\delta(\mathbf{x}_i, \mathbf{x}_j) - \delta(\Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j))\right)^2, \quad (2)$$

where $\delta(\cdot,\cdot)$ is the "distance" between the two points in the appropriate Euclidean space. In other words, MDS seeks a low-dimensional representation of the input data that preserves the pairwise distances as well as possible. Although it is possible to use any distance measures (i.e., metric) as δ, the most common choice by far is the Euclidean distance, e.g., $\delta(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|$ where $\|\cdot\|$ is the ℓ^2-norm. We note that the embedding minimizing J_{MDS} could be nonlinear in general.

Instead of using the distance or dissimilarity, it is also possible to use the similarity among the data points. If the similarity is defined by the centered correlation as

$$\alpha(\mathbf{x}_i, \mathbf{x}_j) := (\mathbf{x}_i - \overline{\mathbf{x}})^T (\mathbf{x}_j - \overline{\mathbf{x}}), \quad \overline{\mathbf{x}} := \frac{1}{m_\star} \sum_{i=1}^{m_\star} \mathbf{x}_i, \quad (3)$$

then instead of Eq. (2) one can consider the embedding minimizing the following

$$J_{\text{CS}}(\Psi) := \sum_{i=1}^{m_\star} \sum_{j=i}^{m_\star} \left(\alpha(\mathbf{x}_i, \mathbf{x}_j) - \alpha\left(\Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j)\right)\right)^2. \quad (4)$$

[1] In the statistics literature, the data matrix is often defined as the transpose of our definition, i.e., each row vector is a training signal.

This is called *Classical MDS* or *Classical Scaling*(CS). Using the matrix-vector notation, and let $\widetilde{X}$ be the centered data matrix, i.e., the mean of the column vectors of X is subtracted from each column of X, which can be written as $\widetilde{X} = XH$ where $H := I - \mathbf{1}\mathbf{1}^T/m_\star$ is the centering matrix of size $m_\star \times m_\star$ and $\mathbf{1} := (1,\dots,1)^T \in \mathbb{R}^{m_\star}$. Let $\widetilde{\Psi}(X) := \Psi(X)H$. (Note that if we assume Ψ is a linear map, then $\widetilde{\Psi}(X) = \Psi(\widetilde{X})$, but let us proceed without assuming this linearity now.) Then, Eq. (4) can be written as

$$J_{\mathrm{CS}}(\Psi) = \left\| \widetilde{X}^T\widetilde{X} - \widetilde{\Psi}(X)^T\widetilde{\Psi}(X) \right\|_F^2, \tag{5}$$

where $\|\cdot\|_F$ is the Frobenius norm. Since $\mathrm{rank}(\Psi(X)) = s$ and $\mathrm{rank}(\widetilde{\Psi}(X)) = s-1$, the minimization of $J_{\mathrm{CS}}(\Psi)$ leads to $\widetilde{\Psi}(X)^T\widetilde{\Psi}(X)$ should be the best rank$(s-1)$-approximation of $\widetilde{X}^T\widetilde{X}$. Let $\widetilde{X} = U\Sigma V^T$ be the singular value decomposition of $\widetilde{X}$. Then, $\widetilde{X}^T\widetilde{X} = V\Sigma^T\Sigma V^T$. Hence, the best rank-$(s-1)$ approximation of $\widetilde{X}^T\widetilde{X}$ is $\sum_{k=1}^{s-1}\sigma_k\mathbf{v}_k\mathbf{v}_k^T$ where σ_k are sorted in the non-increasing order. This implies that

$$\widetilde{\Psi}(X) = \begin{bmatrix} \sigma_1\mathbf{v}_1^T \\ \vdots \\ \sigma_{s-1}\mathbf{v}_{s-1}^T \end{bmatrix} = \Sigma_{s-1}V_{s-1}^T,$$

where $\Sigma_{s-1} = \mathrm{diag}(\sigma_1,\dots,\sigma_{s-1})$ and $V_{s-1} = (\mathbf{v}_1,\dots,\mathbf{v}_{s-1})$. However, because $\mathbf{1} \in \mathbb{R}^{m_\star}$ is an eigenvector of $\widetilde{X}$ corresponding to the eigenvalue 0 (this can be easily seen from $H\mathbf{1} = \mathbf{0}$), the mean of the column vectors of the matrix $\Sigma_k V_k^T$ is $\mathbf{0}$ as long as $\sigma_k \gneqq 0$ i.e., $k \le \mathrm{rank}(\widetilde{X})$ because $V_k^T\mathbf{1} = \mathbf{0}$. In other words, $\Sigma_k V_k^T$ is already centered for each such k. This also implies that we can set $\Psi(X) = \Sigma_s V_s^T$ as our s-dimensional embedding as long as $1 \le s \le \mathrm{rank}(\widetilde{X})$, which is usually the case since $s \ll \min(d, m_\star)$ and the training signals normally linearly independent, i.e., $\mathrm{rank}(\widetilde{X}) = \min(d, m_\star) - 1$.

Now it is clear that the classical scaling is equivalent to the popular Principal Component Analysis (PCA), which was pointed out originally by [15]; see also [19, Sec. 14.3], [7, Sec. 24.1]. Since the first s PCA basis vectors are the eigenvectors of the sample covariance matrix $\widetilde{X}\widetilde{X}^T/m_\star$ corresponding to the largest s eigenvalues, these are the first s column vectors of U, which form U_s. Now, from the SVD of $\widetilde{X}$, we have $\widetilde{X}V_s = U_s\Sigma_s$ or equivalently $U_s = \widetilde{X}V_s\Sigma_s^{-1}$. Now, the first s principal component of the centered data matrix $\widetilde{X}$ is simply the expansion of $\widetilde{X}$ with respect to U_s. Hence, those principal components (or the expansion coefficients) are

$$U_s^T\widetilde{X} = \Sigma_s^{-1}V_s^T\widetilde{X}^T\widetilde{X} = \left[\Sigma_s | O_{s\times(m_\star - s)}\right]V^T = \Sigma_s V_s^T.$$

In other words, this is exactly the same as CS. Hence, we use the term PCA interchangeably with CS or classical MDS in this article, and define

$$\Psi_{\mathrm{PCA}}(X) := \Sigma_s V_s^T, \quad XH = U\Sigma V^T. \tag{6}$$

One of the major drawbacks of PCA and CS is their inability to incorporate *local* geometric information. This is clear from Eq. (3), which gives a small value even if $\mathbf{x}_i$ and $\mathbf{x}_j$ are very close as long as $\mathbf{x}_i - \overline{\mathbf{x}}$ and $\mathbf{x}_j - \overline{\mathbf{x}}$ are almost perpendicular.

2.2 Laplacian eigenmaps

Unlike PCA/CS, Laplacian eigenmaps [2, 3] incorporate *local* geometric information in the ambient space $\mathbb{R}^d$ when constructing a low-dimensional embedding. To explain this more precisely, let us define the weight function $w(\mathbf{x}_i, \mathbf{x}_j)$. A canonical example of this weight function is the Gaussian:

$$w_\varepsilon(\mathbf{x}_i, \mathbf{x}_j) := \mathrm{e}^{-(\delta(\mathbf{x}_i, \mathbf{x}_j)/\varepsilon)^2}, \tag{7}$$

where $\varepsilon > 0$ is the scale parameter and $\delta(\cdot,\cdot)$ is an appropriate metric in the original ambient space $\mathbb{R}^d$. Although there are other choices, we will use the Euclidean distance (i.e., ℓ^2-norm) as δ throughout this article. Such a weight function w_ε defines the notion of a local neighborhood at each point $\mathbf{x} \in \mathscr{X}$ via the affinity between $\mathbf{x}$ and other points in $\mathscr{X}$, and the value of the scale parameter ε specifies the size of this neighborhood. Moreover, as explained in [3], when the dataset $\mathscr{X}$ approximately lies on a lower-dimensional manifold in $\mathbb{R}^d$, using the weights w_ε on the graph constructed from the data points corresponds to an approximation of the heat kernel on this manifold. We also note that choosing the appropriate value of ε for a given training dataset $\mathscr{X}$ is quite important yet difficult in practice. We will discuss more about it in Sec. 4.2.

Now, we seek an embedding map Ψ that minimizes the following objective function

$$J_{\mathrm{LE}}(\Psi) := \sum_{i=1}^{m_\star} \sum_{j=1}^{m_\star} \|\Psi(\mathbf{x}_i) - \Psi(\mathbf{x}_j)\|^2 w_\varepsilon(\mathbf{x}_i, \mathbf{x}_j). \tag{8}$$

Using the matrix-vector notation, this leads to the following optimization problem:

$$\min_{\Psi(X) \in \mathbb{R}^{s \times m_\star}} \mathrm{tr}\left(\Psi(X) L \Psi(X)^T\right) \quad \text{subject to } \Psi(X) D \Psi(X)^T = I, \tag{9}$$

where the matrices are defined as

$$W := (w_\varepsilon(\mathbf{x}_i, \mathbf{x}_j)), \quad D := \mathrm{diag}\left(\sum_j w_\varepsilon(\mathbf{x}_1, \mathbf{x}_j), \ldots, \sum_j w_\varepsilon(\mathbf{x}_{m_\star}, \mathbf{x}_j),\right).$$

The matrices of size $m_\star \times m_\star$, $L := D - W$, D, and W are called the (unnormalized) *graph Laplacian*, the degree matrix (diagonal), and the weighted adjacency matrix, respectively. The constraint $\Psi(X) D \Psi(X)^T = I$ is imposed to prevent Ψ from mapping X to a subspace whose dimension is lower than s. It is now straightforward to show that the minimizer $\Psi(X)$ of Eq. (9) satisfies the following generalized eigen-

value problem:

$$L\Psi(X)^T = D\Psi(X)^T\Lambda,$$

where $\Lambda = \mathrm{diag}(\lambda_0, \lambda_1, \ldots, \lambda_{s-1})$ is a $s \times s$ diagonal matrix listing the eigenvalues in the *non-decreasing* order, i.e., $0 \leq \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_s$. The above generalized eigenvalue problem is equivalent to

$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^T = \Psi_{\mathrm{rw}}(X)^T\Lambda_{\mathrm{rw}}, \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}W. \tag{10}$$

Then, the *Laplacian Eigenmap* of the training dataset $\mathscr{X}$ is defined as $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s\times m_\star}$.

Four important remarks are in order. First, notice that the row vectors of $\Psi_{\mathrm{rw}}(X)$—if they are transposed—form the eigenvectors of L_{rw}. Second, note that $\lambda_0 = 0$ and the corresponding eigenvector is $\mathbf{1} \in \mathbb{R}^{m_\star}$, i.e., $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$, because if we view the data points as nodes and w_ε as the edge weights of a graph, then our graph is *fully connected*. This constant vector $\mathbf{1}$ does not provide any information (except that the graph is fully connected, which we know a priori) and is useless for the embedding. Hence, we solve Eq. (10) for the $(s+1)$-dimensional embedding $\Psi(X)$, and use its 2nd to the last rows as the s-dimensional embedding $\Psi_{\mathrm{rw}}(X)$. Finally, $D^{-1}W$ is now a row-stochastic matrix (i.e., the sum of each row is 1, representing a transition probability distribution of a random walk over the graph). Hence, this multiplication of D^{-1} to L or W from left is called the *row-stochastic normalization*. That is the reason why we used the subscript rw in Eq. (10), which stands for 'random walk'.

Second, instead of Eq. (10), one can consider a different normalization of the graph Laplacian, which is sometimes useful for numerical computation and studying its relationship to the spectral geometry (see, e.g., [8]). This is based on the following eigenvalue problem

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^T = \Psi(X)^T_{\mathrm{sym}}\Lambda_{\mathrm{sym}}, \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}. \tag{11}$$

Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians. These two are related in the following manner:

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

For more about the differences between these two normalizations, see the excellent tutorial paper on the spectral clustering [24].

Third, when the data arise from non-uniform sampling of a manifold, embedding via a Laplacian eigenmap may result in distortion of the manifold when embedded into the reduced space. Such sensitivity to sampling densities may be a serious drawback in certain cases; see [22] for some examples. In fact, this was one of the motivations for Coifman and Lafon [9, 22] to develop a density-invariant normalization of the weights on the graph before computing the graph Laplacian, which we will explain in the next subsection. This normalization would eliminate sensitivity to sampling densities of the Laplacian eigenmaps. However, this normalization was not presented in the original definition of Laplacian eigenmaps [3] and is one of two

important distinctions between the diffusion maps and the Laplacian eigenmaps. (The other is the incorporation of the eigenvalues into the map). Therefore, when we refer to Laplacian eigenmaps in this article, we stand by its original definition. When comparing the performance of different methods in our numerical experiments, we perform density-invariant normalization only when computing diffusion maps.

Fourth, the eigenvalues of the graph Laplacians have been studied extensively for many years; e.g., [8] is a fundamental reference, whose definition of the graph Laplacian is in fact L_{sym}. They have also been applied for a variety of purposes including our own work on characterizing neuronal dendrite patterns [38]. The eigenvectors of the graph Laplacians have also drawn attention recently from computer science and discrete mathematics communities. See e.g., [6] for the basic reference, and [24] for the application to spectral clustering.

2.3 Diffusion maps

Following the work of Coifman and Lafon [9, 22], we construct the diffusion operator on $\mathscr{X}$ as follows. First, we apply the *density-invariant normalization* to the weighted adjacency matrix W, which can be written as

$$\widetilde{W} := D^{-1} W D^{-1}. \tag{12}$$

Then, we apply the row-stochastic normalization as the Laplacian eigenmap algorithm, i.e.,

$$\widetilde{A}_{\text{rw}} := \widetilde{D}^{-1} \widetilde{W}, \tag{13}$$

where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{W}$. Instead of the graph Laplacian L_{rw} used in the Laplacian eigenmaps, the diffusion map algorithm focuses on the normalized weighted adjacency matrix $\widetilde{A}_{\text{rw}}$. One can interpret $\widetilde{A}_{\text{rw}}$ as the *transition matrix* of a random walk on $\mathscr{X}$ or the *diffusion operator* on $\mathscr{X}$. Then, the tth power $\widetilde{A}^t_{\text{rw}}$ corresponds to running such random walk t steps forward in time. The information (i.e., a random walk) propagates more easily and quickly among the regions of high affinity (i.e., data points connected with the larger values of w_ε) than those of low affinity. This is essentially how we capture the local geometric information in the data. The diffusion map algorithm then performs the eigenanalysis

$$\widetilde{A}_{\text{rw}} \Psi_{\text{DM}}(X)^T = \Psi_{\text{DM}}(X)^T \Lambda_{\text{DM}}, \tag{14}$$

where the eigenvalues are sorted in *non-increasing* order. The *diffusion map* is then defined (with a slight abuse of notation) as

$$\Psi^t_{\text{DM}}(X) := \Lambda^t_{\text{DM}} \Psi_{\text{DM}}(X). \tag{15}$$

It is important to point out its relationship with the Laplacian eigenmap (if $\widetilde{W}$ is used for constructing $L_{\rm rw}$ instead of W in Eq. (10)):

$$\Psi^1_{\rm DM}(X) = \Psi_{\rm rw}(X), \quad \Lambda_{\rm DM} = I - \Lambda_{\rm rw}.$$

Hence the largest eigenvalue is $\lambda_0 = 1$ and the corresponding eigenvector is $\mathbf{1} \in \mathbb{R}^{m_\star}$, i.e., $\widetilde{A}_{\rm rw}\mathbf{1} = \mathbf{1}$. Similarly to the Laplacian eigenmaps, this constant vector is useless for the embedding. Hence, we solve Eq. (14) for the $(s+1)$-dimensional embedding $\Psi(X)$, and use its 2nd to the last rows and the eigenvalues $\lambda_1 \geq \cdots \geq \lambda_s$ for $\Psi_{\rm DM}(X)$ in Eq. (15).

Two important remarks are in order here. First, one can also define the symmetric version $\widetilde{A}_{\rm sym} := \widetilde{D}^{-\frac{1}{2}}\widetilde{W}\widetilde{D}^{-\frac{1}{2}}$ in a similar manner as $L_{\rm sym}$ of Eq. (11). In order to compute $\Psi_{\rm DM}(X)$, we can use either: 1) SVD of $\widetilde{A}_{\rm rw}$ and use the 2nd to $(s+1)$st left singular vectors; or 2) symmetric eigenvalue solver for $\widetilde{A}_{\rm sym}$, and then normalize its 2nd to $(s+1)$st eigenvectors by the 1st eigenvector $\widetilde{D}^{\frac{1}{2}}\mathbf{1}$ corresponding to the eigenvalue $\lambda_0 = 1$ (i.e., $\widetilde{A}_{\rm sym}\widetilde{D}^{\frac{1}{2}}\mathbf{1} = \widetilde{D}^{\frac{1}{2}}\mathbf{1}$). For the details on this normalization, see Appendix B of [20].

Second, Coifman and Lafon defined the approximate *diffusion distance* between two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ in the original ambient space is simply the ℓ^2 distance between the embedded points in the diffusion space, i.e.,

$$\delta^t_{\rm DM}(\mathbf{x}_i, \mathbf{x}_j) := \|\Psi^t_{\rm DM}(\mathbf{x}_i) - \Psi^t_{\rm DM}(\mathbf{x}_j)\| = \sqrt{\sum_{k=1}^{s} \lambda_k^{2t}(\psi_k(\mathbf{x}_i) - \psi_k(\mathbf{x}_j))^2},$$

where $\psi_k(\mathbf{x}_i)$ is the (k,i)th entry of the matrix $\Psi_{\rm DM}(X) \in \mathbb{R}^{s\times m_\star}$ following the convention of Eq. (1), which does not include the first eigenvector $\mathbf{1}$.

2.4 Out-of-sample multiscale extension via geometric harmonics

Our goal is in classifying newly obtained unlabeled test ensembles based on a classification rule learned from the labeled training ensembles. In order to make meaningful inference from the training ensembles to the test ensembles, we need to have the same low-dimensional representation for all ensembles. That is, we need to embed test ensembles into the same reduced space as the training ensembles. Hence, it becomes necessary for us to extend the embedding map computed on the training ensembles to the test ensembles. If the embedding map is PCA/CS, then such extension is quite simple: it suffices to expand the centered test data matrix $\widetilde{Y} = YH$ by the left singular vectors U_s of $\widetilde{X}$, i.e., $U_s^T\widetilde{Y}$ is the extension of the test ensembles by PCA. For a nonlinear map such as the Laplacian eigenmaps and the diffusion maps, we employ the multiscale extension scheme proposed in [20], which is based on "geometric harmonics" originally introduced in [22, Chap. 3] and [10]. Let us call this scheme GHME (geometric harmonics multiscale extension) for short. We

now review the GHME scheme using our matrix-vector notation as in the previous subsections.

The GHME scheme is an improvement of *the Nyström extension method* proposed in [14, 5]. First consider the Gaussian kernel matrix defined on the training dataset $\mathscr{X}$ with the scale parameter $\sigma > 0$, which is different from ε used in the weight function in Eq. (7) for constructing the Laplacian eigenmaps and the diffusion maps as follows:

$$W_\sigma(X) := (w_\sigma(\mathbf{x}_i, \mathbf{x}_j)) = \left(\mathrm{e}^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2}\right) \in \mathbb{R}^{m_\star \times m_\star}. \tag{16}$$

Since $W_\sigma(X)$ is positive semi-definite, let us consider the eigenfunction expansion of this kernel matrix:

$$W_\sigma(X) = \boldsymbol{\Phi}^T M \boldsymbol{\Phi}, \quad \boldsymbol{\Phi} := \begin{bmatrix} \phi_1^T \\ \vdots \\ \phi_{m_\star}^T \end{bmatrix} \in \mathbb{R}^{m_\star \times m_\star}, \quad M := \mathrm{diag}(\mu_1, \ldots, \mu_{m_\star}). \tag{17}$$

where the nonnegative eigenvalues $\{\mu_j\}$ are sorted in non-increasing order, and the column vectors of $\boldsymbol{\Phi}^T$, i.e., $\{\phi_1, \ldots, \phi_{m_\star}\}$ form an orthonormal basis for $\ell^2(\mathscr{X}) \subset \mathbb{R}^{m_\star}$, and $\phi_i := (\phi_i(\mathbf{x}_1), \ldots, \phi_i(\mathbf{x}_{m_\star}))^T$. Now, consider the kth eigenpair (μ_k, ϕ_k), i.e., $W_\sigma(X)\phi_k = \mu_k \phi_k$. The ith row of this equality gives us

$$\phi_k(\mathbf{x}_i) = \frac{1}{\mu_k} \sum_{j=1}^{m_\star} w_\sigma(\mathbf{x}_i, \mathbf{x}_j) \phi_k(\mathbf{x}_j).$$

The Nyström extension of ϕ_k from $\mathscr{X}$ to $\mathbf{y} \in \mathscr{Y}$ is defined as

$$\overline{\phi}_k(\mathbf{y}) := \frac{1}{\mu_k} \sum_{j=1}^{m_\star} w_\sigma(\mathbf{y}, \mathbf{x}_j) \phi_k(\mathbf{x}_j). \tag{18}$$

Since the eigenvectors $\{\phi_k\}$ form an orthonormal basis for $\ell^2(\mathscr{X})$, any function $\mathbf{f} := (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{m_\star}))^T \in \ell^2(\mathscr{X})$ (e.g., $\mathbf{f}^T$ can be any row of the Laplacian eigenmap $\boldsymbol{\Psi}_{\mathrm{rw}}(X)$ or the diffusion map $\boldsymbol{\Psi}^t_{\mathrm{DM}}(X)$) can be expanded as

$$\mathbf{f} = \sum_{k=1}^{m_\star} \langle \mathbf{f}, \phi_k \rangle \, \phi_k,$$

and the jth entry of both sides gives us

$$f(\mathbf{x}_j) = \sum_{k=1}^{m_\star} \langle \mathbf{f}, \phi_k \rangle \, \phi_k(\mathbf{x}_j).$$

Thus the Nyström extension of f from $\mathscr{X}$ to $\mathbf{y} \in \mathscr{Y}$ can be defined as

$$\overline{f}(\mathbf{y}) := \sum_{k=1}^{m_\star} \langle \mathbf{f}, \phi_k \rangle \, \overline{\phi}_k(\mathbf{y}).$$

In order to understand what we have done here, let us plug Eq. (18) in the righthand side of the above equation.

$$\begin{aligned} \overline{f}(\mathbf{y}) &= \sum_{k=1}^{m_\star} \frac{\langle \mathbf{f}, \phi_k \rangle}{\mu_k} \sum_{j=1}^{m_\star} w_\sigma(\mathbf{y}, \mathbf{x}_j) \phi_k(\mathbf{x}_j) \\ &= \sum_{k=1}^{m_\star} \frac{\phi_k^T \mathbf{f}}{\mu_k} w_\sigma(\mathbf{y}, :) \phi_k \\ &= w_\sigma(\mathbf{y}, :) \boldsymbol{\Phi}^T M^{-1} \boldsymbol{\Phi} \mathbf{f}, \end{aligned} \tag{19}$$

where we borrowed the convenient notation of MATLAB® [2], i.e.,

$$w_\sigma(\mathbf{y}, :) := (w_\sigma(\mathbf{y}, \mathbf{x}_1), \ldots, w_\sigma(\mathbf{y}, \mathbf{x}_{m_\star})) \in \mathbb{R}^{1 \times m_\star}$$

.

We observe that the range of the extension in Eq. (18) is proportional to σ. If the ratio $\|\mathbf{y} - \mathbf{x}_j\|/\sigma$ is large for all $\mathbf{x}_j \in \mathscr{X}$, then $\overline{\phi}_k(\mathbf{y})$ will be numerically small and hence may not be meaningful. Hence the extension scale σ should be as large as possible. On the other hand, for large enough σ, the kernel matrix $W_\sigma(X)$ defined in Eq. (16) becomes ill-conditioned, i.e., μ_k tends to 0 more quickly compared to the case of small σ. Thus the Nyström extension in (18) will blow up. Furthermore, it is well known hat the extension range depends on the smoothness of the function to be extended [22, Chap. 3], [10]. If f is fairly smooth, it can be extended far away from the training set while it has limited extension range if f varies wildly on $\mathscr{X}$. We can clearly see these problems in Eq. (19): it truly boils down to the ill-conditionedness of $W_\sigma(X)$. In fact, if we can compute M^{-1} without blowing up, i.e., $\mu_{m_\star} \gneqq 0$, then $\boldsymbol{\Phi}^T M^{-1} \boldsymbol{\Phi} = W_\sigma(X)^{-1}$ because of Eq. (17). Moreover, by setting $\mathbf{y} = \mathbf{x}_j \in \mathscr{X}$ in the righthand side of Eq. (19) recovers $f(\mathbf{x}_j)$:

$$w_\sigma(\mathbf{x}_j, :) \boldsymbol{\Phi}^T M^{-1} \boldsymbol{\Phi} \mathbf{f} = w_\sigma(\mathbf{x}_j, :) W_\sigma(X)^{-1} \mathbf{f} = \mathbf{e}_j^T \mathbf{f} = f(\mathbf{x}_j),$$

where $\mathbf{e}_j \in \mathbb{R}^{m_\star}$ is the jth standard basis vector in $\mathbb{R}^{m_\star}$. In practice, however, $W_\sigma(X)$ is ill-conditioned, and we need to truncate M^{-1} to the first $p \times p$ submatrix and $\boldsymbol{\Phi}$ to the first p rows where p satisfying $1 \leq p < m_\star$ must be appropriately chosen. Let M_p^{-1}, $\boldsymbol{\Phi}_p$ be these truncated matrices. Then, Eq. (19) can be approximated without blowup:

$$\begin{aligned} \overline{f}(\mathbf{y}) &\approx w_\sigma(\mathbf{y}, :) \boldsymbol{\Phi}_p^T M_p^{-1} \boldsymbol{\Phi}_p \mathbf{f} \\ &= w_\sigma(\mathbf{y}, :) W_{\sigma,p}(X)^\dagger \mathbf{f}, \end{aligned} \tag{20}$$

[2] MATLAB is a registered trademark of The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760-2098.

where $W_{\sigma,p}(X)^\dagger := \Phi_p^T M_p^{-1} \Phi_p$ is the *pseudoinverse* of $W_\sigma(X)$ using their top p singular values and vectors. Hence, if we want to extend a low-dimensional embedding map $\Psi(X) = [\psi_1|\cdots|\psi_s]^T$, we have

$$\begin{aligned}\overline{\Psi}(\mathbf{y}) = \begin{bmatrix}\overline{\psi}_1(\mathbf{y})\\ \vdots \\ \overline{\psi}_s(\mathbf{y})\end{bmatrix} &\approx \left\{w_\sigma(\mathbf{y},:)\Phi_p^T M_p^{-1}\Phi_p[\psi_1|\cdots|\psi_s]\right\}^T \\ &= \left\{w_\sigma(\mathbf{y},:)\Phi_p^T M_p^{-1}\Phi_p\Psi(X)^T\right\}^T \\ &= \Psi(X)\Phi_p^T M_p^{-1}\Phi_p w_\sigma(:,\mathbf{y}) \\ &= \Psi(X)W_{\sigma,p}(X)^\dagger w_\sigma(:,\mathbf{y}),\end{aligned}$$

where $w_\sigma(:,\mathbf{y}) := (w_\sigma(\mathbf{x}_1,\mathbf{y}),\ldots,w_\sigma(\mathbf{x}_{m_\star},\mathbf{y}))^T = w_\sigma(\mathbf{y},:)^T \in \mathbb{R}^{m_\star\times 1}$.

The GHME scheme determines this rank p of the pseudoinverse by the following:

$$p = \underset{1\le k\le m_\star}{\arg\max}\left\{\frac{\mu_1}{\mu_k} \le \eta\right\}. \tag{21}$$

where $\eta > 0$ is some fixed condition number. In other words, p is the largest possible stable rank of $W_\sigma(X)$, which is bounded from above by η. Clearly, if one sets η too large, $p = m_\star$ may occur. The extension $\overline{f}$ in Eq. (20) is well-defined on $\mathscr{X}\cup\mathscr{Y}$, but it is not equal to f on the training set $\mathscr{X}$ unless $p = m_\star$ and σ is set so small that $W_\sigma(X)$ has a stable inverse. Such a case, however, is not of our interest because setting σ too small practically disconnects data points in $\mathscr{X}$. In fact as $\sigma\to 0$, $W_\sigma(X)\to I$ as long as $\mathbf{x}_i \ne \mathbf{x}_j$ for all $i\ne j$ in $\mathscr{X}$. Yet observe that if the value of σ decreases, the eigenvalues $\mu_k \to 0$ more slowly. This allows us to use larger p in Eq. (20), making $\overline{f}$ a better approximation of f on $\mathscr{X}$. Based on this observation, the GHME iteratively searches for an extension $\overline{f}$ that approximates f on $\mathscr{X}$ with an preset error tolerance $\rho > 0$ by slowly decreasing the value of the extension scale σ. The GHME scheme can be summarized as follows:

Algorithm 2.1 (The GHME of Lafon, Keller, and Coifman[20]) *Suppose f is a function defined on the training set $\mathscr{X}$ and to be extended to a new dataset $\mathscr{Y}$.*

Step 1: Fix a condition number $\eta > 0$ and an error tolerance $\rho > 0$. Set the extension scale $\sigma = \sigma_0$ for some large value σ_0.

Step 2: Compute eigenvalues $\{\mu_k\}$ and orthonormal eigenvectors $\{\phi_k\}$ of the Gaussian kernel matrix $W_\sigma(X)$ and expand f (on the training set $\mathscr{X}$) in this eigenbasis.

Step 3: On the training set $\mathscr{X}$, approximate f by $\overline{f}$ defined in Eq. (20) *by finding p in Eq.* (21)*. Then compute the approximation error*

$$Err := \left(\sum_{k>p} |\langle f,\phi_k\rangle|^2\right)^{1/2}.$$

If $Err > \rho$, *set* $\sigma \leftarrow \frac{1}{2}\sigma$ *and return to Step 2. Otherwise, continue.*

Step 4: *Using the value of p obtained in Step 3, compute the final approximate extension defined in Eq.* (20) *for each* $\mathbf{y} \in \mathscr{Y}$*:*

$$\bar{f}(\mathbf{y}) \approx w_\sigma(\mathbf{y},:)\Phi_p^T M_p^{-1}\Phi_p \mathbf{f} = w_\sigma(\mathbf{y},:)W_{\sigma,p}(X)^\dagger \mathbf{f}.$$

3 Earth Mover's Distance

Once our training ensembles are embedded in a lower-dimensional space, say $\mathbb{R}^s$ (normally $1 \le s \ll \min(d, m_\star)$), via the diffusion map or Laplacian eigenmaps so as the test ensembles are via the GHME scheme as reviewed in the previous section, we are ready to classify the latter. To do so, we need to quantify a discrepancy (or measure a "distance") between a test ensemble and each training ensemble in $\mathbb{R}^s$. One of the simplest ideas such as the sum of the pairwise Euclidean distances (in the embedding space) between the data points in one ensemble and those in the other ensemble will not work because: 1) there may be a few outliers in those ensembles that ruin such a distance measure; and 2) the number of data points (signals) in each ensemble may be different in general so that the simple sum of the distances may lead to an erroneous label assignment.

Another idea is to use the Hausdorff distance (HD), which was used by Lafon, Keller, and Coifman as the ensemble distance measure [20]. The HD between any two ensembles $\mathscr{X}, \mathscr{Y} \subset \mathbb{R}^s$ is defined as

$$d_H(\mathscr{X}, \mathscr{Y}) := \max\left(\max_{\mathbf{y}\in\mathscr{Y}} \min_{\mathbf{x}\in\mathscr{X}} \|\mathbf{x}-\mathbf{y}\|, \max_{\mathbf{x}\in\mathscr{X}} \min_{\mathbf{y}\in\mathscr{Y}} \|\mathbf{x}-\mathbf{y}\|\right), \tag{22}$$

where $\|\cdot\|$ denotes the Euclidean distance in $\mathbb{R}^s$. As our numerical experiments in Sec. 5 demonstrates, also as one can easily imagine from this definition, the HD is still quite sensitive to the outliers.

The above considerations have led us to adopt the *Earth Mover's Distance* (EMD) as a discrepancy/distance measure between ensembles, which is more robust and more suitable in our classification problems than the other measures. In this section, we briefly review the key aspects of the EMD.

The definition of EMD is based on the solution to a discrete *optimal mass transportation problem*. EMD represents the minimum cost of moving earth (or sand) from some source locations to fill up holes at some sink locations. In other words, given any two mass (or probability) distributions, one of them can be viewed as a distribution of earth and the other a distribution of holes, then the EMD between the two distributions is the minimum cost of rearranging the mass in one distribution to obtain the other. In the continuous setting, this problem is known as the *Monge-Kantorovich optimal mass transfer* problem and has been well studied over the past 100 years (see [13] for an introductory reading on the problem). The importance here is that EMD can be used to measure the discrepancy between two multidimensional distributions.

In the discrete setting, the optimal mass transfer problem can be formulated as a linear optimization problem as follows [33, 34]. Suppose we want to quantify the discrepancy between two ensembles of signals (or feature vectors) of length $s \in \mathbb{N}$. Suppose all the signals in one ensemble are grouped into m clusters using a clustering algorithm such as the K-means algorithm (see Sec. 4.1 for what we actually use to construct a signature in our experiments; for the details of various clustering algorithms, see e.g., [17, Chap. 14]) and this ensemble is represented by a set of m cluster centers and the associated weights (member population within each cluster) as $P = \{(\hat{\mathbf{x}}_1, p_1), \cdots, (\hat{\mathbf{x}}_m, p_m)\}$. Note that one can optionally normalize the weights p_i's so that $\sum_i p_i = 1$, which allows us to view P as a probability distribution of the signals in $\mathscr{X}$ over $\mathbb{R}^s$. Suppose the other ensemble $\mathscr{Y}$ is represented, using the same procedure, by a set of n cluster centers and weights $Q = \{(\hat{\mathbf{y}}_1, q_1), \cdots, (\hat{\mathbf{y}}_n, q_n)\}$. Such P and Q are called the *signatures* of the ensembles. Viewing P as a source (earth) distribution and Q as a sink (hole) distribution, we can now formulate the discrete optimal mass transport problem leading to the EMD as a measure of the discrepancy between two signatures P and Q.

Suppose the cost of moving one unit of mass from $\hat{\mathbf{x}}_i$ to $\hat{\mathbf{y}}_j$ is $c(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_j)$, and f_{ij} denotes the amount of mass flow from $\hat{\mathbf{x}}_i$ to $\hat{\mathbf{y}}_j$. There are many possibilities in defining this cost, but we use $(1/2)\|\hat{\mathbf{x}}_i - \hat{\mathbf{y}}_j\|^2$, i.e., the half of the *squared* Euclidean distance, which gives more preference to closely placed points than the usual Euclidean distance between them. Then, the transportation cost is defined as

$$\mathrm{COST}(P, Q, \mathbf{F}) := \sum_{i=1}^{m} \sum_{j=1}^{n} c(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_j) f_{ij},$$

where $\mathbf{F} := [f_{ij}] \in \mathbb{R}^{m \times n}$. The optimal mass transfer problem seeks the flow $\mathbf{F}^*$ that transfers the maximum allowable amount of earth to fill up the holes with minimum total transportation cost, i.e.,

$$\mathbf{F}^* = \arg\min_{\mathbf{F} \in S} \mathrm{COST}(P, Q, \mathbf{F}),$$

where $\mathbf{F} \in S \subset \mathbb{R}^{m \times n}$ means that $\mathbf{F}$ must satisfy the following set of constraints:

(i) $f_{ij} \geq 0$, for all i, j ;
(ii) $\sum_{j=1}^{n} f_{ij} \leq p_i$, for all $1 \leq i \leq m$;
(iii) $\sum_{i=1}^{m} f_{ij} \leq q_j$, for all $1 \leq j \leq n$; and
(iv) $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{m} p_i, \sum_{j=1}^{n} q_j\right)$.

These constraints ensure that: (i) one can only move earth from P to Q, not vice versa; (ii) the amount of earth moved from P is no more than the sum of the weights p_i; (iii) the amount of earth received at Q is no more than the sum of the weights q_j; and (iv) the maximum allowable amount of earth is moved.

Once the optimal flow $\mathbf{F}^*$ from P to Q is found, EMD is then defined as the total cost normalized by the total flow:

$$\mathrm{EMD}(P,Q) := \frac{\mathrm{COST}(P,Q,\mathbf{F}^*)}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^*} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} c(\hat{\mathbf{x}}_i,\hat{\mathbf{y}}_j) f_{ij}^*}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}^*}. \tag{23}$$

Notice that the normalization factor is the total weight of the smaller signature due to the constraint (iv). This normalization ensures that smaller signatures are not favored in the case when two signatures have different total weights. Furthermore, EMD is symmetric, i.e., $\mathrm{EMD}(P,Q) = \mathrm{EMD}(Q,P)$ for any two distributions P and Q. Notice also that each signature may consists of different number of clusters or "bins" in the reduced embedding space as indicated by m and n above. Since there is no guarantee to have the same number of cluster centers in the signature for each ensemble, EMD is more suitable for our classification problem compared to the other metrics. Finally, notice that the EMD directly depends on P and Q, which are not uniquely determined from the ensembles $\mathscr{X}$ and $\mathscr{Y}$. In other words, the EMD depends on the clustering algorithm to construct P and Q. Hence, it requires care to construct the signatures, which we will discuss in detail in Sec. 4.1.

4 An Algorithm for Signal Ensemble Classification using Low-Dimensional Embeddings with Earth Mover's Distance

In this section, we will describe our proposed algorithm to classify test signal ensembles $\mathscr{Y}^1,\ldots,\mathscr{Y}^N$ given training signal ensembles $\mathscr{X}^1,\ldots,\mathscr{X}^M$ using a low-dimensional embedding and the EMD as the ensemble distance measure. Then, we will describe how to construct appropriate signatures of ensembles in the embedding space. Finally, we will discuss an important practical issue, i.e., how to select several key tuning parameters in our algorithm.

We summarize our proposed method for signal ensemble classification in the following algorithm.

Algorithm 4.1 (Signal Ensemble Classification via a Low-Dimensional Embedding and EMD) *Let $\mathscr{X} = \bigcup_{i=1}^{M} \mathscr{X}^i$, $\mathscr{Y} = \bigcup_{j=1}^{N} \mathscr{Y}^j$ be the training and test ensembles, respectively.*

Step 1. Training Stage using $\mathscr{X}$:

i. Preset a large enough initial dimension $1 < s_0 < \min(d, m_\star)$ of the embedding space.
ii. Choose Ψ from $\{\Psi_{\mathrm{PCA}}, \Psi_{\mathrm{rw}}, \Psi_{\mathrm{sym}}, \Psi_{\mathrm{DM}}^1\}$, construct a low-dimensional embedding map $\Psi : \mathscr{X} \subset \mathbb{R}^d \to \mathbb{R}^{s_0}$, and embed $\mathscr{X}$ into the temporary reduced space in $\mathbb{R}^{s_0}$.
iii. For $i = 1 : M$, construct a signature P^i using $\Psi(X^i) \in \mathbb{R}^{s_0}$.
iv. Determine the appropriate reduced dimension $1 \le s \le s_0$.
v. For $i = 1 : M$, adjust the signature P^i in the final reduced space $\mathbb{R}^s$.

Step 2. Test Stage for $\mathscr{Y}$:

i. Extend the learned map Ψ in Step 1 to the test ensembles $\mathscr{Y}$.

ii. *For $j = 1:N$, construct the signature Q^j of $\mathscr{Y}^j$.*

iii. *For $j = 1:N$, compute $\{\mathrm{EMD}(P^i, Q^j)\}_{i=1}^M$ and find $i_j := \arg\min_{1\le i\le M} \mathrm{EMD}(P^i, Q^j)$. Assign the test ensemble $\mathscr{Y}^j$ the label of $\mathscr{X}^{i_j}$. That is,* ***assign a label by the 1-nearest neighbor classifier with EMD as its base distance measure****.*

We will now describe what are important for implementing and running the above algorithm for given signal ensembles.

4.1 Signature construction and embedding dimension estimation

The above overall algorithm did not show the detailed differences between the case when one chooses Ψ_{PCA} and the cases of the nonlinear embedding maps, i.e., Ψ_{rw}, Ψ_{sym}, and $\Psi^1_{\mathrm{DM}}\}$. In this section, we will describe these differences in detail.

Let us first describe Step 1-iii, i.e., the signature construction step. If we use PCA as Ψ, then we use the standard K-means algorithm to construct the clusters in the top s_0 PCA coordinates of $\mathscr{X}^i$. For the K-means algorithm, it is essential to determine the number of clusters K_i for each training ensemble $\mathscr{X}^i$. There are many possibilities for determining the number of optimal clusters (see e.g., [17, Sec. 14.3.11]), out of which we use a heuristic yet simple method called the "elbow" criterion. This can be explained as follows. As we increase the number of clusters, the fitness (or within-cluster dissimilarity) function, which is the sum of all point-to-centroid distances, will decrease very rapidly at first then slowly. This will result in a "kink"or "elbow" in the plot of the fitness function versus the number of clusters. Hence, we choose the number of clusters at the elbow as the natural candidate for the number of clusters to form for the given data.

On the other hand, if we use the Laplacian eigenmaps or the diffusion map as Ψ, then we use the *elongated* K-means (*ekmeans*) algorithm [39] to determine the number of clusters in the signature for each training ensemble $\mathscr{X}^i$. The idea of *ekmeans* was adapted from the spectral clustering algorithm proposed in [30] by replacing the Euclidean distance with an elongated distance in the computation of point-to-center distances. In [39], the elongated distance (e-dist) between two points $\mathbf{x}, \mathbf{c} \in \mathbb{R}^{s_0}$ is defined as

$$\text{e-dist}(\mathbf{x},\mathbf{c}) := (\mathbf{x}-\mathbf{c})^T M_\alpha (\mathbf{x}-\mathbf{c}), \quad M_\alpha := \frac{1}{\alpha}(I - P_{\mathbf{c}}) + \alpha P_{\mathbf{c}}, \quad P_{\mathbf{c}} := \frac{\mathbf{c}\mathbf{c}^T}{\mathbf{c}^T\mathbf{c}}. \quad (24)$$

As one can see from this definition, the e-dist considers two components of the vector $\mathbf{x}-\mathbf{c}$: 1) the radial component $P_{\mathbf{c}}(\mathbf{x}-\mathbf{c})$, i.e., the orthogonal projection of $\mathbf{x}-\mathbf{c}$ onto the direction $\mathbf{c}$; and 2) the traversal component $(I-P_{\mathbf{c}})(\mathbf{x}-\mathbf{c})$, i.e., the complementary projection of $\mathbf{x}-\mathbf{c}$ with respect to $\mathbf{c}$. The parameter α controls the balance between these two components: the smaller α becomes, the more traversal direction is emphasized in the distance computation. In all of our numerical experiments below, we set $\alpha = 0.2$, the value recommended by the authors in [39]. The use of

the e-dist allows us to group points lying inside a thin elongated ellipsoid to form a cluster, as opposed to inside a sphere.

Ekmeans exploits the geometric properties of the eigenvectors of the graph Laplacian matrix or the adjacency matrix to cluster the data and automatically determine the number of intrinsic clusters. It starts the clustering process in the top 2 eigenspace with three initialized centers: two centers at two different elongated clusters and one at the origin of the eigenspace. If there are more than two elongated clusters, the center at the origin will be dragged to a cluster not yet accounted for. Then the algorithm moves the clustering process to the top 3 eigenspace, creating three cluster centers and adds a fourth center at the origin. This process is repeated until no additional cluster is found. This clustering process stops at the top K eigenspace if there are K (intrinsic) clusters in the data.

Step 1-iv determines the appropriate dimension s of the final reduced embedding space. In the case of $\Psi = \Psi_{\text{PCA}}$, s is determined by the decay of the singular values σ_k of the centered training data matrix $\widetilde{X}$. In fact, using PCA, we can combine Steps 1-iii, iv, and v into a single step by first estimating the effective dimension s by the decay of the singular values, then construct a signature in $\mathbb{R}^s$ for each $\mathscr{X}^i$ using the K-means algorithm with the elbow criterion. By this way, we do not need to readjust the signatures in Step 1-v. As for the Laplacian eigenmaps and diffusion maps, suppose the *ekmeans* algorithm finds an intrinsic dimension K_i for each training ensemble $\mathscr{X}^i$. This number K_i is also the intrinsic number of clusters in that ensemble. This intrinsic number of K_i clusters does not change when the ensemble $\mathscr{X}^i$ is embedded into an eigenspace of dimension greater than K_i. This is because no additional cluster other than the K_i elongated clusters already accounted for will be found even if $\mathscr{X}^i$ is embedded into such a larger eigenspace. Therefore, it is natural to set the dimension of the reduced embedding space for the training ensembles $\mathscr{X}$ to be the maximum of K_i over all $i = 1, \ldots, M$, i.e., $s := \max_{1 \le i \le M} K_i$.

Step 1-v is necessary for both the Laplacian eigenmaps and the diffusion map as Ψ because of the following reason. The cluster centers for ensemble $\mathscr{X}^i$ determined by *ekmeans* in Step 1-iii are vectors in the top K_i-dimensional subspace of the temporary reduced space $\mathbb{R}^{s_0}$. Step 1-iv determines the final dimension s of the reduced space. In order for us to use EMD in Step 2, however, all the cluster centers must be in $\mathbb{R}^s$. Hence, to bring all cluster centers into $\mathbb{R}^s$, we need to re-cluster $\mathscr{X}^i$'s embedded points in $\mathbb{R}^s$ by running the standard K-means with the e-dist as its base distance measure to reform the K_i clusters. Here, we use the previous cluster memberships as a starting condition for the standar K-means. At the end of this re-clustering process, all cluster centers of the signature for each training ensemble are in the final reduced space $\mathbb{R}^s$.

Finally, in Step 2-i, we extend the learned map to the test ensembles. If $\Psi = \Psi_{\text{PCA}}$, then this extension can be computed simply by $U_s^T \widetilde{Y}$ If $\Psi = \Psi_{\text{rw}}$, Ψ_{sym}, or Ψ_{DM}, then we use the GHME Algorithm 2.1.

4.2 On tuning parameters

Other than the dimensionality s of the reduced embedding space, there are three more tuning parameters to be determined in our proposed method: the scale ε for the weights $w_\varepsilon(\mathbf{x}_i, \mathbf{x}_j)$ defined in Eq. (7); the error tolerance ρ in approximating the extension of the nonlinear embedding maps; and the cutoff bound η for the condition number of the extension kernel. Clearly we have to select each of these parameters wisely. We describe now how to tune these parameters.

4.2.1 Determination of ε

The scale $\varepsilon > 0$ for Gaussian weights $w_\varepsilon(\mathbf{x}, \mathbf{y})$ should be chosen so that the corresponding weighted graph is numerically connected. Connectedness of the graph guarantees the existence and uniqueness of the eigenvector corresponding to the smallest eigenvalue 0 of L_{rw} in Eq. (10) and the largest eigenvalue 1 of $\widetilde{A}_{\text{rw}}$ in Eq. (14), and therefore of the stationary distribution of the Markov random walk process on the graph. Thus, when computing diffusion maps, the value for ε must be large enough to ensure that every point in the graph is connected to at least one other point. It is clear, however, that any affinity or dissimilarity between the data points is obscured if ε is too large since w_ε converges to 1 regardless of its arguments (i.e., every data point is equally similar to any other point) as ε increases to infinity. One suggestion given in [24] is to choose ε *"in the order of the mean distance of a point to its kth nearest neighbor"* (where k is proportional to the logarithm of the number of points in the graph). In our numerical experiments below, we select ε to be the mean of the Euclidean distances from each point to its k-nearest neighbors. We determine this k by running cross-validation experiments on the training set $\mathscr{X}$. We start with $k = 1$, run a cross-validation trial, and record the results. Next, continue to increment k by Δk and run a cross-validation trial until the classification results stop to improve. When this happens, the previous value for k, say $k^\star$, may already be optimal. To ensure that this $k^\star$ is indeed optimal, however, we decrease Δk to $\lfloor \Delta k/2 \rfloor$ and search for possibly better value for k around $k^\star$ if we set Δk large. The initial size of Δk depends on the total number of the training signals $m_\star$. In our lip-reading example, we start with $\Delta k = 5$ since the training set is large ($990 \le m_\star \le 1204$). However, in the classification of underwater objects, $m_\star$ is small ($96 \le m_\star \le 120$), so we start with $\Delta k = 1$. As an illustration, in our lip-reading example below, the average value for $k^\star$ determined by this cross-validation search method (over 100 experimental trials) turned out to be 16 when EMD was used for the ensemble distance measure and 14 with HD. These are approximately 1.4% to 1.6% of the training signals $m_\star$. For the underwater object classification, any values of $k^\star \in [2, 30]$ with EMD and $k^\star \in [8, 40]$ with HD produced the same optimal classification results. These $k^\star$'s correspond to about 2% to 40% of $m_\star$. We note that when k is, say, 5% of the number of the training signals, the corresponding Gaussian-weighted graph has approximately 5% percent of its edges weighted by a weight greater than or equal to $1/\mathrm{e}$, and the rest have smaller weights. In other

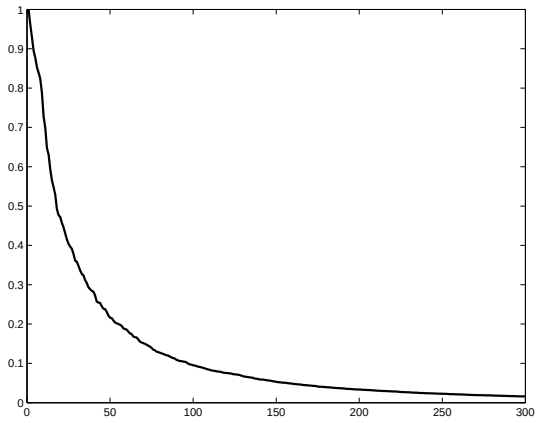

Fig. 1 The largest 300 eigenvalues of the diffusion kernel in one trial of the lip-reading experiments ($\varepsilon = 657$).

words, the graph is sparse but not too sparse. With this choice of ε, the spectrum of the diffusion kernel decays relatively fast. In Fig. 1 we plot the largest 300 eigenvalues of the normalized diffusion matrix $\widetilde{A}_{\mathrm{rw}}$ defined in Eq. (13) of one trial of the lip-reading experiments. The value of ε is 657 corresponding to $k^\star = 16$. We see that the eigenvalues decrease quickly. Fast decay of the spectrum implies that any random walk initiated on the graph converges quickly to steady state and that the diffusion distance can be approximated more accurately with a smaller number of eigenfunctions. In other words, we should be able to detect clustering behaviors in the data with a small number of time steps t for such a case. In the numerical experiments in Sec. 5, we used $t = 1$ in the diffusion maps defined in Eq. (15). Setting $t = 1$ and optimizing ε is certainly an easier strategy compared to the simultaneously seeking the optimal combination of $t > 1$ and ε although the latter may generate better classification results. Also, there may be datasets on which some value of $t > 1$ results in better classification than setting $t = 1$. At least for those datasets in Sec. 5, however, having found the value ε appropriate for the data while setting $t = 1$ was enough for identifying grouping patterns.

We note that the value of ε or k found to be optimal for the diffusion maps may not be optimal for the Laplacian eigenmaps, and vice versa. Thus, we need to determine the optimal k separately for each case. We could start at $k = 1$ and increment by Δk as described above. However, if we have already found the optimal value $k^\star$ for the diffusion map (or the Laplacian eigenmap) and if we want to save the computational time to find the optimal value k for the Laplacian eigenmap (or the diffusion map), then we could proceed as follows. We start the search with $k = k^\star$; then we increase k by e.g., $\Delta k = 5$ and proceed as described above. If the classification is not improving, we also need to search in the opposite direction, i.e., decrease k by $\Delta k = 5$ and proceed (in decreasing direction) as described above. In our numerical experiments below, we computed the diffusion map first. Therefore, when searching for the optimal k for the Laplacian eigenmap, we took the approach described in this paragraph. The optimal value k found when computing the Laplacian eigenmaps (the random walk version) is 60 (EMD) and 67 (HD) for the lip-reading experiments

(averaging over 100 experimental trials) while in the underwater object classification any values of $k \in [12,40]$ (EMD) and $k = 7$ or 13 (HD) generated the optimal classification results. For the symmetric version of the Laplacian eigenmaps, the average optimal k was 57 (EMD) and 62 (HD) for the lip-reading experiments while any values of $k \in [3,8]$ (EMD) and $k \in [1,8]$ (HD) were optimal for the underwater object classification.

4.2.2 Determination of ρ

When choosing a value for the error tolerance ρ for the approximation of the out-of-sample extension of the nonlinear embedding maps in Algorithm 2.1 that is used in Step 2-i of our Algorithm 4.1, we should keep in mind that a small error bound ρ corresponds directly to a small extension scale σ as we discussed in Sec. 2.4. Suppose we know a priori that the function we want to extend is fairly smooth on our training ensembles $\mathscr{X}$. Then we can expect the extension to have a large extension range. In this case, we can be greedy and set ρ small, i.e., we want the extension $\overline{\Psi}$ to be very close to Ψ on $\mathscr{X}$. A heuristic value to set for ρ is 1% of the number of the training signals $m_\star$. This gives an average of 0.01 bound on the error at each point where the approximation is being computed.

4.2.3 Determination of η

To determine a cutoff threshold η for the condition number of the Gaussian extension kernel matrix $W_\sigma(X)$ in Eq. (16), we have to keep in mind the approximation error tolerance ρ. If ρ is small, then η has to be large. In addition, as σ increases, the condition number of the kernel matrix also increases. To predict how large η might get, we can take advantage of Step 1 of our Algorithm 4.1. Let κ be the condition number of $\widetilde{A}_{\text{rw}}$, $\widetilde{A}_{\text{sym}}$, L_{rw}, or L_{sym} with the parameter ε optimally chosen in Step 1. It is easy to show that the condition number of $W_\sigma(X)$ is proportional to κ if $\sigma = \varepsilon$. Furthermore, since we usually set the initial $\sigma_0 > \varepsilon$ in Algorithm 2.1, the condition number of the initial Gaussian kernel is larger than κ. Hence we can consider setting η larger than κ and inversely proportional to ρ. In our numerical experiments, we set $\eta = \min(\kappa, 10^5)/\rho$.

5 Numerical Experiments and Results

We now illustrate how our proposed algorithm can be applied to signal ensemble classification problems where the data characterizing each object consist of ensembles of signals instead of a single signal. We will show two examples of application. The first example is classification of underwater objects via analyzing Synthetic Aperture Sonar (SAS) waveforms reflected from the objects. The second example

is a lip-reading application in which we identify the spoken word from a sequence of video frames extracted from a silent video segment. As we mentioned earlier, we use PCA, Laplacian eigenmaps (two different normalizations L_{rw} and L_{sym}), and diffusion map. We will also compare the performance of EMD with that of the Hausdorff distance (HD) defined in Eq. (22) in our classification problems.

5.1 Classification of underwater objects

The data in this example were provided by the Naval Surface Warfare Center, Panama City (NSWC-PC), FL. They were collected from three different controlled experiments in a fresh water test pond at NSWC-PC. For details of the experiments, see [29]. In each of the three experiments, two objects were placed — either buried in the sand or proud — at the bottom of the pond. In each experiment, one object was a sphere made of an iron casing filled with a different material, and the other object was a solid aluminum cylinder of different length. A sinusoidal pulse was transmitted across the floor of the pond and the reflected signals were recorded over a period of time sampled at uniform rate. The data obtained contain waveforms reflected from the entire area of the pond floor. Waveforms corresponding to objects are extracted and preprocessed using the algorithms described in [26, Chap's. 3, 4], which is an improved version of the algorithm presented in [27]. This yields ensembles of waveforms per object, and each ensemble consists of rectangular blocks of waveforms.

Our goal is to identify objects according to their material compositions regardless of their shapes. We name the sphere and the cylinder in Experiment j as Sj and Cj, for $j = 1,2,3$. Sphere $S1$ was filled with air, so we categorize it as one class with label **IA** for iron-air. Spheres $S2$ and $S3$ were filled with silicone oil so we group them into another class with label **IS** for iron-silicone. All three cylinders were of the same diameter and of the same material, so we grouped them into one class with label **Al** for aluminum. We note, however, that the physical length of $C1$ and that of $C2$ were the same while that of $C3$ was slighly shorter than the half of $C1$ and $C2$.

These waveform data are of extremely high dimension: each data point $\mathbf{x}_i$ is a rectangular block of waveforms (i.e., 2D array) of size 17 (cross-range samples) by 600 (time samples), i.e., $d = 17 \times 600$; see Fig. 4 for some examples. As for the number of ensembles, we have six ensembles corresponding to these six objects, Sj, Cj, $j = 1,2,3$. The number of data points (blocks of waveforms) contained in each ensemble was 8, 8, 16 for $S1$, $S2$, $S3$, respectively while that of the three cylinders was 32 each. We set aside one ensemble of waveforms (corresponding to one object) as a test ensemble and trained our algorithm on the remaining five sets. For example, when we use the $S1$ ensemble as a test ensemble, we have, using our notation, $\mathscr{X} = \bigcup_{i=1}^{5} \mathscr{X}^i$, $\mathscr{Y} = \mathscr{Y}^1$, i.e., $M = 5$ and $N = 1$ while the number of data points (each of which is of size 17×600) in each ensemble is, $m_1 = 8$, $m_2 = 16$, $m_3 = m_4 = m_5 = 32$, and $n_1 = 8$. Then we applied the steps in Algorithm 4.1 to classify the test object. We cycled through all six objects, that is, we repeated the classification process six

Table 1 Identification of Underwater Objects

Object		$C1$	$C2$	$C3$	$S1$	$S2$	$S3$
True Label		**Al**	**Al**	**Al**	**IA**	**IS**	**IS**
PCA	EMD	**Al**	**Al**	**Al**	**IS**	**IS**	**IA**
	HD	**Al**	**Al**	**Al**	**IS**	**IS**	**IA**
LE_{rw}	EMD	**Al**	**Al**	**Al**	**Al**	**IS**	**IS**
	HD	**Al**	**Al**	**Al**	**Al**	**Al**	**IS**
LE_{sym}	EMD	**Al**	**Al**	**Al**	**Al**	**IS**	**IS**
	HD	**Al**	**Al**	**Al**	**Al**	**IS**	**IS**
DM	EMD	**Al**	**Al**	**Al**	**Al**	**IS**	**IS**
	HD	**Al**	**Al**	**Al**	**Al**	**IS**	**IS**

times. The classification results for all six runs are shown in Table 1. Using EMD coupled with nonlinear dimensionality reduction methods (DM, LE_{rw}, LE_{sym}), we consistently and correctly classify all three cylinders as objects of class **Al** and the spheres $S2$ and $S3$ as objects of class **IS**. Moreover, the mistake of labeling the sphere $S1$ as **Al** is also consistent. Note that this error is expected since the class **IA** contains only one member ensemble $S1$. We have no training data for this class when the sphere $S1$ is left out as test data. Furthermore, note that having $S1$ as part of the training data does not confuse the classification of the spheres $S2$ and $S3$ when one of the nonlinear dimensionality reduction methods is applied. This is not the case when PCA is used for dimensionality reduction. We will discuss more on this phenomenon in Sec. 6 below.

Classification of the objects using HD is less consistent among the different dimensionality reduction/embedding methods. The main reason for this is that HD is highly sensitive to outliers. For a closer look, let us examine the distribution of the points embedded by the Laplacian eigenmap (the random walk version) into the lower-dimensional space. Fig. 2 shows three sets of points embedded into the first three coordinates of LE_{rw} computed from the training data consisting of all three cylinders and the spheres $S1$ and $S3$. The sphere $S2$ is first left out as test data then embedded into the same reduced embedding space via the GHME scheme. In this figure, blue crosses correspond to cylinder $C3$ (class **Al**), green triangles correspond to sphere $S3$ (class **IS**), and red circles correspond to the unlabeled test object $S2$ (true label is **IS**). Black stars are the cluster centers, i.e., the representatives, in the signature of each object. We see that the circles (corresponding to object $S2$) are on average close to the triangles (corresponding to object $S3$), but because of the points along the long tail of $S3$, the HD between $S2$ and $S3$ turns out to be larger than that between $S2$ and $C3$. The actual EMD and HD values are shown in Table 2. Note that the dimension s of the reduced embedding space is actually 12 not 3 in this case. We can see from Table 2 that the smallest EMD value is 0.0053 corresponding to

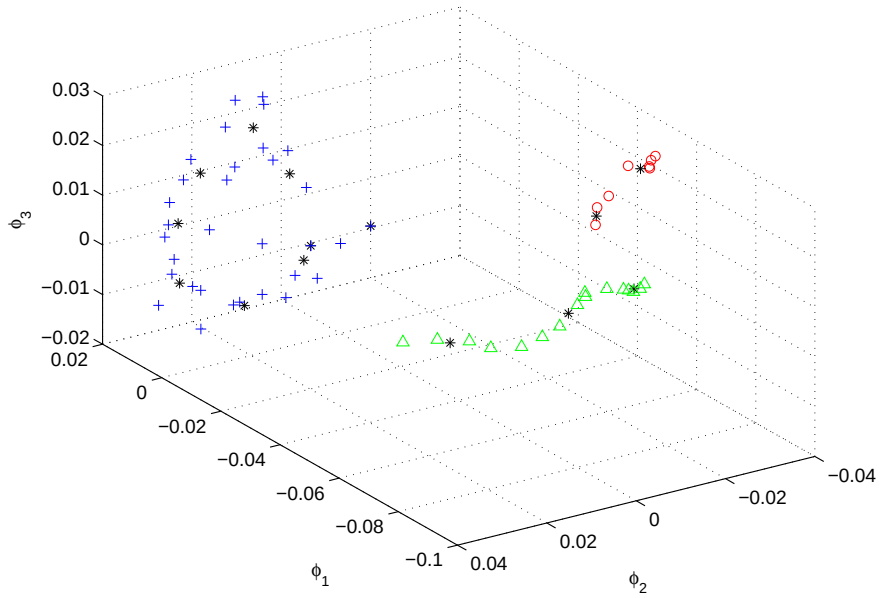

Fig. 2 Three underwater objects in the first three $\mathrm{LE}_{\mathrm{rw}}$ coordinates. $C3$ (*crosses*), $S3$ (*triangles*), unlabeled test object $S2$ (*circles*). Stars indicate the cluster centers. The Laplacian eigenmap in this case was computed from the training data consisting of all three cylinders and the spheres $S1$ and $S3$.

Table 2 EMD and HD values in the $\mathrm{LE}_{\mathrm{rw}}$ coordinates between sphere object $S2$ and all other objects. ($k = 13$ for this example)

Object	$C1$	$C2$	$C3$	$S1$	$S3$
EMD	0.0070	0.0064	0.0057	0.0085	0.0053
HD	0.1917	0.2374	0.1237	0.1500	0.1684

$S3$ and the smallest HD value is 0.1237 corresponding to $C3$. Thus, EMD correctly labels object $S2$ as **IS**, but HD mislabels $S2$ as **Al**.

5.2 Lip reading experiment

In this section, we present our results on a simplified version of the lip-reading problem to illustrate how our proposed algorithm can be applied in practice. The objective of lip reading is to train a machine to automatically recognize the spoken words from the movements of the lips captured on silent video segments (no sound is involved). Much research effort has been devoted to this area. Many published algorithms involve sophisticated feature selection. In this example, we simply per-

Fig. 3 Examples of the preprocessed video frames used in our lip-reading experiments. Both the top row and the bottom row are the subsets of the video frames speaking the digit 'one'. Note that the bottom row corresponds to the situation when the subject spoke 'one' while smiling.

Table 3 Lip-Reading total recognition errors (averaged over 100 experimental trials)

PCA EMD	PCA HD	LE_{rw} EMD	LE_{rw} HD	LE_{sym} EMD	LE_{sym} HD	DM EMD	DM HD
5.3%	9.4%	36.1%	36.1%	26.0%	27.6%	24.1%	25.2%

form dimensionality reduction on the sequences of images (i.e., video frames). We do not extract any particular lip features from the images unlike those in [31, 40] and many other publications. Furthermore, the lips data we use were collected from one speaker. It may be necessary to extract and use more sophisticated features when more speakers, i.e., more variations in the lips, are involved.

We recorded a subject speaking the first five digits ('one',...,'five') ten times using a Nikon Coolpix digital camera sampling at a rate of 60 frames per second. We then extracted the video frames from each movie clip and did the following simple preprocessing. First, we convert the images from color to gray scales ranging from 0 to 255. Then we cropped each image to a 55×70 pixels window around the lips to compensate for translations. (The speaker's nose was marked with a color marker to facilitate automatic cropping of the video frames). Figure 3 shows subsets of video frames of two such movie clips. For each spoken digit, we randomly selected five video sequences from the available ten such sequences in our collection as training ensembles. This gave us a total of 25 video sequences as the training ensembles and another 25 for test ensembles. Hence, using our notation, our experimental setting for each run is: $\mathscr{X} = \bigcup_{i=1}^{25} \mathscr{X}^i$, $\mathscr{Y} = \bigcup_{j=1}^{25} Y^j$, i.e., $M = N = 25$, while $30 \leq m_i, n_i \leq 63$ (i.e., the number of video frames in each sequence varies between 30 and 63), and the dimension of each data point (i.e., a video frame) is $d = 55 \times 70$.

We applied Algorithm 4.1 to classify the test video sequences. We repeated the whole process 100 times. The total misclassification rates (averaging over 100 experimental trials) are shown in Table 3. Again, we see that using EMD gives smaller recognition errors than using HD except the LE_{rw} case where the results tied. Table 3 shows high classification errors for the Laplacian eigenmaps, in particularly, LE_{rw} while those of DM are at least more than 10% better. Compared to these nonlinear

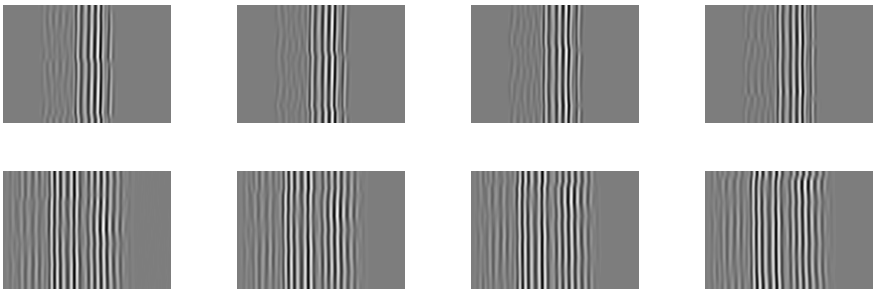

Fig. 4 Selected waveforms corresponding to the sphere $S1$ (*top row*) and sphere $S3$ (*bottom row*) in the underwater object experiment. The horizontal axis represents time (600 time samples) whereas the vertical axis indicates the 17 cross-range coordinates.

dimensionality reduction methods, however, PCA, i.e., the linear method, worked much better for this dataset, i.e., 5.3% with EMD and 9.4% with HD.

6 Discussion and Conclusion

We have seen that PCA performed better than the local nonlinear methods in the lip-reading experiment whereas the situation was opposite in the underwater objects classification experiment. The reason is because the characterizing patterns in the lip-reading problem are 'global' whereas those in the underwater-objects problem are 'local'. To explain this in more details, suppose we view a video sequence by eyes. In order for us to determine what the spoken word actually is, we need to see how the lips move throughout the entire video sequence. We do not care so much that the shape of the lips in each individual video frame is slightly different from what we have seen in the past. As long as the dynamics (or total movements) of the lips in this video sequence are similar to what we remember, we would be able to recognize the spoken word. In other words, the *trajectory* of the video frames in the embedded space is decisive. Recognizing and discrimininating such trajectories is a 'global' pattern recognition problem. We believe that is the reason why PCA in the lip-reading experiment outperformed the diffusion maps and Laplacian eigenmaps.

On the other hand, the sonar waveform blocks of the same class are quite homogeneous. To illustrate this, we display in Fig. 4 some selected samples of waveforms reflected from the spheres $S1$ and $S3$ in the underwater-objects experiment. Recall that each rectangular block is viewed as one data point. We can see clearly that the blocks belonging to sphere $S1$ are quite different from those from sphere $S3$ while our eyes can barely discern the differences between the blocks within the same class. Local nonlinear methods can map each class to the tighter 'localized' clusters and enhance the between-class differences compared to PCA. This is the reason why these local nonlinear methods outperformed PCA for this example.

In conclusion, we have proposed an algorithm for classifying objects that are characterized or described by ensembles of signals using a low-dimensional embedding and the Earth Mover's Distance (EMD). Our algorithm sets up the framework for application of EMD to such classification problems. We have shown that EMD is more robust to noise and hence more appropriate for discrimination of ensembles than the Hausdorff distance.

We have provided two examples of practical applications for our proposed algorithm. The lip-reading application is of global-pattern nature, therefore dimensionality reduction by PCA proved successful. On the other hand, the classification of underwater objects is of local-pattern nature, thus dimensionality reduction by nonlinear local methods such as diffusion maps and Laplacian eigenmaps gave better results than PCA.

We did not incorporate the discriminant information during the training stage explicitly in this article. For example, we have not compared the performance of our proposed method with the other typical classification strategy, i.e., extracting discriminant features via e.g., Local Discriminant Basis [36, 37] followed by setting up a classifier, e.g., Linear Discriminant Analysis [17, Chap. 4], Support Vector Machines [17, Chap. 12], etc., on the extracted feature vectors. We plan to investigate the embedding techniques explicitly incorporating the discriminant information during the training stage. Some of our attempts along this direction can be found in [26].

Acknowledgments

This work was partially supported by the ONR grants N00014-06-1-0615, N00014-07-1-0166, N00014-09-1-0041, N00014-09-1-0318, the NSF grant DMS-0410406, and the NSF VIGRE grants DMS-0135345, DMS-0636297. A preliminary version of a subset of the material in this article was presented at the SPIE Wavelets XII Conference, in San Diego, in August 2007 [23]. We thank Dr. Quyen Huynh and Dr. Joe Lopes of NSWC-PC for providing the experimental sonar data. We have used the SPECTRAL Toolbox version 0.1 distributed on the web by Dr. Guido Sanguinetti and Dr. Jonathan Laidler to compute Elongated Kmeans. Finally, we would like thank Dr. Bradley Marchand for his help in processing the sonar data used in the numerical experiments and Mr. Julien van Hout for his help in numerical experiments on underwater object classification.

References

1. Basseville, M.: Distance measures for signal processing and pattern recognition. Signal Processing **18**(4), 349–369 (1989)
2. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: T.K. Leen, T.G. Dietterich, V. Tresp (eds.) Advances in Neural Information Pro-

cessing Systems, vol. 13, pp. 585–591. The MIT Press, Cambridge, MA (2001)
3. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation **15**(6), 1373–1396 (2003)
4. Belkin, M., Niyogi, P.: Towards a theoretical foundation for Laplacian-based manifold methods. J. Comput. Syst. Sci. **74**(8), 1289–1308 (2008)
5. Bengio, Y., Paiement, J.F., Vincent, P., Delalleau, O., Roux, N.L., Ouimet, M.: Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. In: S. Thrun, L.K. Saul, B. Schölkopf (eds.) Advances in Neural Information Processing Systems, vol. 16, pp. 177–184. The MIT Press, Cambridge, MA (2004)
6. Bıyıkoğlu, T., Leydold, J., Stadler, P.F.: Laplacian Eigenvectors of Graphs, *Lecture Notes in Mathematics*, vol. 1915. Springer-Verlag, New York (2007)
7. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications, 2nd edn. Springer, New York (2005)
8. Chung, F.R.K.: Spectral Graph Theory. No. 92 in CBMS Regional Conference Series in Mathematics. Amer. Math. Soc., Providence, RI (1997)
9. Coifman, R.R., Lafon, S.: Diffusion maps. Applied and Computational Harmonic Analysis **21**(1), 5–30 (2006)
10. Coifman, R.R., Lafon, S.: Geometric harmonics. Applied and Computational Harmonic Analysis **21**(1), 32–52 (2006)
11. Cover, T.M., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inform. Theory **IT-13**, 21–27 (1967)
12. Donoho, D.L., Grimes, C.: Hessian eigenmaps: Locally linear embedding technique techniques for high-dimensional data. P. Natl. Acad. Sci. USA **100**(10), 5591–5596 (2003)
13. Evans, L.C.: Partial differential equations and Monge-Kantorovich mass transfer. Tech. rep., Dept. Math., Univ. California, Berkeley (2001)
14. Fowlkes, C., Belongie, S., Chung, F., Malik, J.: Spectral grouping using the Nyström method. IEEE Trans. Pattern Anal. Machine Intell. **26**(2), 214–225 (2004)
15. Gower, J.C.: Some distance properties of latent root and vector methods used in multivariate analysis. Biometrika **53**(3/4), 325–338 (1966)
16. Haker, S., Zhu, L., Tannenbaum, A., Angenent, S.: Optimal mass transport for registration and warping. Intern. J. Comput. Vision **60**(3), 225–240 (2004)
17. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edn. Springer-Verlag (2009)
18. Kirby, M., Sirovich, L.: Application of the Karhunen-Loève procedure for the characterization of human faces. IEEE Trans. Pattern Anal. Machine Intell. **12**(1), 103–108 (1990)
19. K.V.Mardia, J.T.Kent, J.M.Bibby: Multivariate Analysis. Academic Press, San Diego, CA (1979)
20. Lafon, S., Keller, Y., Coifman, R.R.: Data fusion and multicue data matching by diffusion maps. IEEE Trans. Pattern Anal. Machine Intell. **28**(11), 1784–1797 (2006)
21. Lafon, S., Lee, A.B.: Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning and data set parameterization. IEEE Trans. Pattern Anal. Machine Intell. **28**(9), 1393–1403 (2006)
22. Lafon, S.S.: Diffusion maps and geometric harmonics. Ph.D. thesis, Dept. Math., Yale Univ. (2004). Downloadable from `http://www.math.yale.edu/~sl349`
23. Lieu, L., Saito, N.: Automated discrimination of shapes in high dimensions. In: D. Van De Ville, V.K. Goyal, M. Papadakis (eds.) Wavelets XII, Proc. SPIE 6701 (2007). Paper # 67011V
24. von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
25. van der Maaten, L.J.P., Postma, E.O., van den Herik, H.J.: Dimensionality reduction: A comparative review. Technical Report TiCC-TR 2009-005, Tilburg Centre for Creative Computing, Tilburg Univ. (2009)
26. Marchand, B.: Local signal analysis for classification. Ph.D. thesis, Dept. Math., Univ. California, Davis (2010)
27. Marchand, B., Saito, N., Xiao, H.: Classification of objects in synthetic aperture sonar images. In: Proc. 14th IEEE Workshop on Statistical Signal Processing, pp. 433–437. IEEE (2007)

28. Murase, H., Nayar, S.K.: Visual learning and recognition of 3d objects from appearance. Intern. J. Comput. Vision **14**(1), 5–24 (1995)
29. Nesbitt, C.L., Lopes, J.L.: Subcritical detection of an elongated target buried under a rippled interface. In: Oceans '04, MTS/IEEE Techno-Ocean '04, vol. 4, pp. 1945–1952 (2004)
30. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: T. Dietterich, S. Becker, Z. Ghahramani (eds.) Advances in Neural Information Processing Systems, vol. 14, pp. 849–856. The MIT Press, Cambridge, MA (2002)
31. Patterson, E.K., Gurbuz, S., Tufekci, Z., Gowdy, J.N.: Moving-talker, speaker-independent feature study, and baseline results using the CUAVE multimodal speech corpus. EURASIP J. Appl. Signal Process. **11**, 1189–1201 (2002)
32. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
33. Rubner, Y., Tomasi, C.: Perceptual Metrics for Image Database Navigation. Kluwer Academic Publishers, Boston, MA (1999)
34. Rubner, Y., Tomasi, C., Guibas, L.J.: The Earth Mover's Distance as a metric for image retrieval. Intern. J. Comput. Vision **40**(2), 99–121 (2000)
35. Saito, N.: Image approximation and modeling via least statistically dependent bases. Pattern Recognition **34**, 1765–1784 (2001)
36. Saito, N., Coifman, R.R.: Local discriminant bases and their applications. J. Math. Imaging Vis. **5**(4), 337–358 (1995). Invited paper
37. Saito, N., Coifman, R.R., Geshwind, F.B., Warner, F.: Discriminant feature extraction using empirical probability density estimation and a local basis library. Pattern Recognition **35**(12), 2841–2852 (2002)
38. Saito, N., Woei, E.: Analysis of neuronal dendrite patterns using eigenvalues of graph Laplacians. JSIAM Letters **1**, 13–16 (2009). Invited paper
39. Sanguinetti, G., Laidler, J., Lawrence, N.D.: Automatic determination of the number of clusters using spectral algorithms. In: Proc. 15th IEEE Workshop on Machine Learning for Signal Processing, pp. 55–60 (2005)
40. Zhang, X., Mersereau, R.M.: Lip feature extraction towards an automatic speechreading system. In: Proc. 2000 International Conference on Image Processing, vol. 3, pp. 226–229. IEEE (2000)
41. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. SIAM J. Sci. Comput. **26**(1), 313–338 (2005)
42. Zhou, S.K., Chellappa, R.: From sample similarity to ensemble similarity — Probabilistic distance measures in reproducing kernel Hilbert space. IEEE Trans. Pattern Anal. Machine Intell. **28**(6), 917–929 (2006)