Chapter 2
Spectral Dimensionality Reduction

Abstract In this chapter a common mathematical framework is provided which forms the basis for subsequent chapters. Generic aspects are covered, after which specific dimensionality reduction approaches are briefly described.

Keywords Spectral dimensionality reduction algorithms · Kernel methods · Spectral graph theory.

Before addressing the open problems it is important to have an understanding of the problem domain itself along with the techniques that have been proposed to perform spectral dimensionality reduction. To fully understand and appreciate the open problems they need to be described in terms of a common mathematical framework. By doing so the problems described in the latter sections can be coherently addressed in relation to a common frame of reference.

This section begins by providing a general mathematical setting within which both spectral dimensionality reduction, and the associated open problems, can be described. Then key algorithms, both linear and nonlinear, are briefly described so as to provide an important point of reference and discussion for the later discussion of open problems.

2.1 A General Setting for Spectral Dimensionality Reduction

To effectively analyse spectral dimensionality reduction and the associated problems it is useful to frame the methodology within a general setting. As the name suggests, at the heart of spectral dimensionality reduction is the spectral decomposition of a square symmetric feature matrix. Different techniques can be distinguished based on the construction of this feature matrix and the eigenvectors that are subsequently used (i.e. smallest or largest). This feature matrix aims to capture certain properties of the

H. Strange and R. Zwiggelaar, *Open Problems in Spectral Dimensionality Reduction*, SpringerBriefs in Computer Science, DOI: 10.1007/978-3-319-03943-5_2,

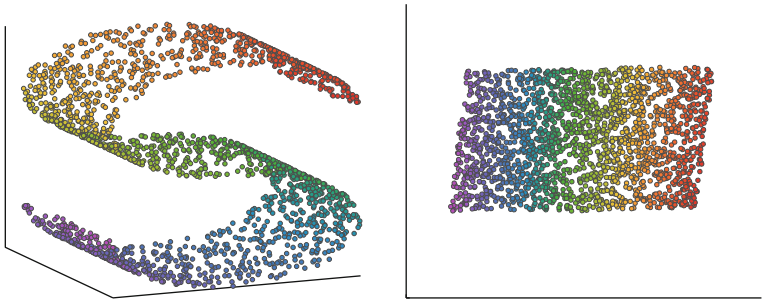

Fig. 2.1 The S-Curve dataset (**a**) along with its low-dimensional embedding (**b**)

subspace or submanifold upon which the data lies, the spectral decomposition of this matrix then gives rise to the low-dimensional embedding (Fig. 2.1). With this in mind the general setting of spectral dimensionality reduction can be defined as follows:

Definition 1 *Given a set of D-dimensional data* $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^D$ *that is sampled from a d-dimensional manifold* $\mathscr{M}$ *such that* $\mathbf{X} \subset \mathscr{M}$*, the goal of dimensionality reduction is to recover a set of d-dimensional data* $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n \in \mathbb{R}^d$ *(*$d \ll D$*) such that* $\mathbf{Y}$ *is a faithful representation of* $\mathbf{X}$ *and preserves certain properties of* $\mathscr{M}$*.*

Due to the vagueness of this definition, many questions naturally present themselves. For example, what constitutes a faithful representation of the original data? Is the high-dimensional data expected to lie on or near a low-dimensional subspace or a low-dimensional submanifold? How is the data sampled from this subspace or submanifold? These types of questions underlie the general assumptions that different approaches make about the setting within which dimensionality reduction takes place and will be returned to in due course.

A more formal definition of spectral dimensionality reduction can be obtained by "filling in" some of the gaps found in Definition 1. As previously mentioned, a feature matrix is built from $\mathbf{X}$ that aims to capture certain properties of the data and will often represent subspace or submanifold properties. Given the original data $\mathbf{X}$, the feature matrix $\mathbf{F}$ is built such that

(i) $\mathbf{F}$ is a square $n \times n$ matrix. Here n could refer to the number of objects in the dataset or the ambient dimensionality of the data
(ii) $\mathbf{F}$ is symmetric, i.e. $\mathbf{F}_{ij} \equiv \mathbf{F}_{ji} \quad \forall \; i, j \in [1, \ldots n]$
(iii) $\mathbf{F}$ is positive semi-definite, that is, $\mathbf{u}^T\mathbf{F}\mathbf{u} \geq 0$ for every $\mathbf{u} \in \mathbb{R}^D$

It is this similarity matrix that distinguishes various spectral dimensionality reduction techniques. For example, $\mathbf{F}$ could measure the covariance of $\mathbf{X}$ as in Principal Components Analysis [1], or the geodesic interpoint distances as in Isomap [2].

Once $\mathbf{F}$ has been built, it is recast in terms of its eigenvectors and eigenvalues using eigendecomposition. By decomposing $\mathbf{F}$ in such a manner the low-dimensional

representation can be found. Given that $\mathbf{F}$ is a square symmetric matrix, it can be recast as

$$\mathbf{F} = \mathbf{Q}\Lambda\mathbf{Q}^T \tag{2.1}$$

where $\mathbf{Q}$ is a $D \times n$ matrix containing the eigenvectors as columns and Λ is a $n \times n$ matrix with the eigenvalues on the diagonal, assuming that $\mathbf{F}$ is of size $n \times n$. Each eigenvector $\mathbf{u}_i \in \mathbf{Q}$ has a corresponding eigenvalue $\lambda_i \in \Lambda$ for which $\mathbf{F}\mathbf{u}_i = \lambda_i \mathbf{u}_i$. The low-dimensional representation $\mathbf{Y}$ is then found by utilising either the top or bottom eigenvectors of the decomposition of $\mathbf{X}$ found using Eq. (2.1).

2.2 Linear Spectral Dimensionality Reduction

Linear approaches to spectral dimensionality reduction make the assumption that the data lies on or near a low-dimensional subspace. In such cases, linear spectral dimensionality reduction methods seek to 'learn' the basis vectors of this low-dimensional subspace so that the input data can be projected onto the linear subspace. The two main methods for linear spectral dimensionality reduction, Principal Components Analysis and Multidimensional Scaling, are both described in this section. Although more powerful nonlinear approaches have been presented in recent years, these linear techniques are still widely used and are worthy of attention since they provide the basis for some of the subsequent nonlinear spectral dimensionality reduction algorithms.

2.2.1 Principal Components Analysis (PCA)

PCA [1, 3] seeks to find the low-dimensional subspace within the data that maximally preserves the covariance up to rotation. This maximum covariance subspace encapsulates the directions along which the data varies the most. Therefore, projecting the data onto this subspace can be thought of as projecting the data onto the subspace that retains the most information. An example embedding found using PCA is shown in Fig. 2.2a.

The first step of PCA is to calculate the $D \times D$ covariance matrix of $\mathbf{X}$,

$$\mathbf{F} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T \tag{2.2}$$

with the assumption that $\mathbf{X}$ is centred at the origin (i.e. zero mean). PCA is the only spectral dimensionality reduction technique where the feature matrix $\mathbf{F}$ is not in terms of the number of data points n, but rather the original dimensionality D. The eigendecomposition of $\mathbf{F}$, found according to Eq. (2.1), gives rise to the basis vectors

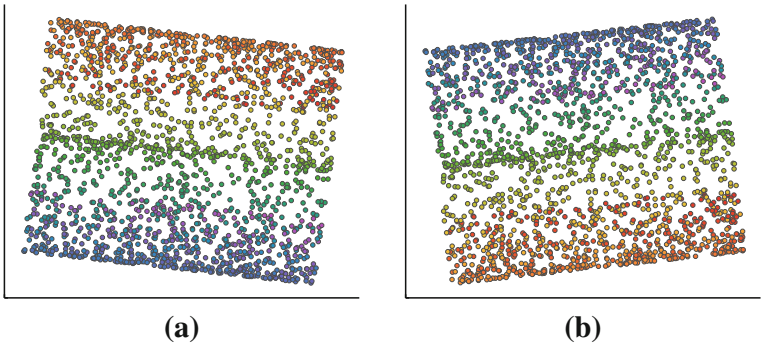

Fig. 2.2 Example 2-dimensional embeddings of the S-Curve dataset found by **a** Principal Components Analysis, and **b** Multidimensional Scaling

of the subspace upon which the data lies on or near. Therefore, the low-dimensional embedding found according to PCA is given by the projection $\mathbf{Y} = \mathbf{X}\mathbf{Q}_{1\ldots d}$, where $\mathbf{Q}_{1\ldots d}$ is the matrix of d eigenvectors ordered in descending order of their associated eigenvalues.

The intuition behind PCA is that the largest eigenvector of the matrix $\mathbf{F}$ corresponds to the dimension in the high-dimensional space along which $\mathbf{X}$ varies the most. Similarly, the second largest eigenvector corresponds to the dimension with the second most variation, and so on. So the top d-eigenvectors describe the d-dimensional subspace which contains the most variance.

2.2.2 Classical Multidimensional Scaling (MDS)

Classical MDS [4], sometimes referred to as metric MDS, shares many similar properties to PCA, as shown by the example low-dimensional embedding in Fig. 2.2b. Whereas PCA seeks to find the subspace that maximally preserves variance, MDS seeks to preserve pairwise distances in the low-dimensional space. As such, MDS takes as input a matrix, $\mathbf{S}$, of squared pairwise distances:

$$\mathbf{S}_{ij} = ||\mathbf{x}_i - \mathbf{x}_j||^2 \tag{2.3}$$

The squared distance matrix in its raw form is not positive semi-definite, so cannot be used as the feature matrix for spectral dimensionality reduction. Therefore, it needs to be converted to a Gram, or inner-product, matrix through the following transformation:

$$\mathbf{F} = -\frac{1}{2}\mathbf{HSH} \tag{2.4}$$

where $\mathbf{H}$ is a centring matrix such that $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{e}\mathbf{e}^T$ where $\mathbf{I}$ is the $n \times n$ identity matrix, and $\mathbf{e}$ is an n vector of all ones. The spectral decomposition of $\mathbf{F}$ gives rise to the top d eigenvalues $\{\lambda_j\}_{j=1}^{d}$ and eigenvectors $\{\mathbf{q}_j\}_{j=1}^{d}$. The low-dimensional embedding found according to MDS is then

$$\mathbf{Y} = \left\{\sqrt{\lambda_j}\mathbf{q}_{ji}\right\}_{i=1}^{n} \tag{2.5}$$

Both PCA and Classical MDS give rise to the same low-dimensional embedding and the Gram matrix (Eq. 2.4) has the same rank and eigenvalues up to a constant factor as the feature (covariance) matrix of PCA [5].

2.3 Nonlinear Spectral Dimensionality Reduction

The central limitation of linear approaches to dimensionality reduction is that they assume the data lies on or near a linear subspace. In practice this may not always be the case; spaces may be locally linear, but unlike the assumption made by linear techniques, globally they may be highly nonlinear. As such, using linear techniques in such circumstances could lead to distorted results with curved areas of the data being projected on top of each other.

Nonlinear spectral dimensionality reduction techniques seek to alleviate this problem by modelling the data not using a subspace, but a submanifold. The data is assumed to be sampled from a low-dimensional manifold embedded within a high-dimensional space. This manifold could be highly nonlinear and convoluted, yet nonlinear methods seek to maintain this manifold structure in the low-dimensional space such that points that are close on the manifold are close in the low-dimensional space, and conversely, points that are far away on the manifold are mapped as far away in the low-dimensional space. An example of this is shown in Fig. 2.3.

In all cases, the feature matrix is built from a graph whose vertices correspond to the input data, and whose edge set corresponds to neighbourhood relations. Therefore, nonlinear spectral dimensionality reduction methods build on the principles laid out in linear algebra and graph theory whereby a graph is reconstructible from its spectrum and the eigenvectors of the graph's adjacency matrix [6].

This section reviews some of the most popular methods for nonlinear spectral dimensionality reduction. This list of methods is by no means exhaustive, rather, the methods included in this section are chosen for their didactic value and also their popularity. Each method corresponds to an important and different paradigm to spectral dimensionality reduction; as such, they are each landmarks within the landscape of spectral dimensionality reduction and provide a brief but sufficient survey of the main trends within this area.

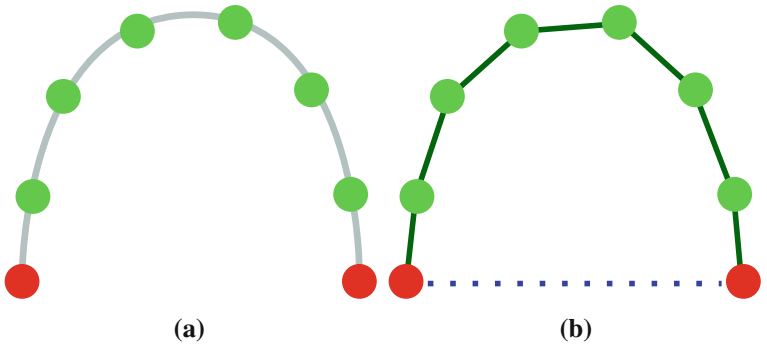

Fig. 2.3 Points sampled from a simple horseshoe shaped manifold (**a**). The two distances in (**b**) show the difference between distances as measured across the manifold and the Euclidean distance. The two end points are connected by the *dotted line* according to the Euclidean distance. However, their manifold distance would be the sum of inter-point distances on the path between the two points. For nonlinear spectral dimensionality reduction techniques, the manifold distances should be used so that the two end points are mapped as far away in the low-dimensional space

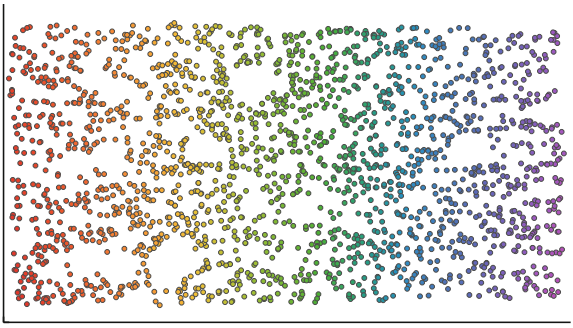

Fig. 2.4 Example 2-dimensional embeddings of the S-Curve dataset found by Isomap with $k = 12$

2.3.1 Isomap

Isomap [2], one of the first true nonlinear spectral dimensionality reduction methods, extends metric MDS to handle nonlinear manifolds. Whereas metric MDS measures inter-point Euclidean distances to obtain a feature matrix, Isomap measures the inter-point *manifold* distances by approximating geodesics. The use of manifold distances can often lead to a more accurate and robust measure of distances between points so that points that are far away according to manifold distances, as measured in the high-dimensional space, are mapped as far away in the low-dimensional space (Fig. 2.3). An example low-dimensional embedding of the S-Curve dataset (Fig. 2.1) found using Isomap is given in Fig. 2.4.

At the heart of Isomap is the computation of the manifold inter-point distances which is achieved by estimating the geodesic distances across a neighbourhood graph. The geodesic distance between two points is represented as $\mathbf{S}_{ij} = \phi(\mathbf{x}_i, \mathbf{x}_j)$, where

$\phi(\cdot)$ is the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ as measured using geodesics. The key insight of Isomap is that the geodesic distance function corresponds to the summation of the distances of 'short-hops' along a neighbourhood graph. Given a graph $G = \langle V, E \rangle$ such that V is the vertex set equal to $\mathbf{X}$, and the edge set E contains the local connectivity of the vertices (calculated by the k-nearest neighbour rule or the ε-neighbourhood rule) the distance $\phi(\mathbf{x}_i, \mathbf{x}_j)$ corresponds to the shortest path in G between vertices $\mathbf{x}_i$ and $\mathbf{x}_j$ as calculated by an algorithm such as Dijkstra's method [7].

As with MDS, the matrix $\mathbf{S}$ is not positive semi-definite, so a Gram matrix is derived according to Eq. (2.4). Once the geodesic distances have been computed, Isomap follows the same algorithm as MDS. The Gram matrix is decomposed into eigenvalues and eigenvectors and the low-dimensional embedding is given by Eq. (2.5). However, unlike MDS, the feature matrix captures manifold distances as opposed to squared Euclidean distances. Therefore, the low-dimensional embedding will maintain manifold properties rather than linear subspace properties.

2.3.2 Maximum Variance Unfolding (MVU)

Sometimes referred to as semidefinite embedding, MVU [8] seeks to find the low-dimensional representation by 'unrolling' the high-dimensional data. As with Isomap, MVU constructs a k-nearest neighbourhood graph to represent the connectivity of the manifold as sampled by the data in $\mathbf{X}$. However, unlike Isomap, spectral decomposition is not performed on this connectivity matrix, rather, the datapoints are unfolded, throughout the formulation of an optimisation problem, by separating the data as much as possible subject to specific constraints. At its core, MVU seeks to maximise the Euclidean distances between the datapoints whilst leaving the distances at a local scale unchanged. This is done by formulating the central optimisation problem as that of a semidefinite program [9].

The solution to the maximum variance unfolding problem is found by constructing a Gram matrix, $\mathbf{F}$, whose top eigenvectors give rise to the low-dimensional representation of the data. MVU seeks to maximise $\sum_{i=1}^{n} \|\mathbf{y}_i - \mathbf{y}_j\|^2$, with $\mathbf{y}_{i,j} \in \mathbf{Y}$, subject to the following constraints

$$(1)\ \|\mathbf{y}_i - \mathbf{y}_j\|^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2\ \forall\ E(i, j) \in G$$
$$(2)\ \sum_i \bar{\mathbf{y}}_i = 0$$

where $E(i, j)$ indicates an edge in the graph G between vertex i and vertex j and so only enforces a constraint on local distances, Constraint (2) ensures the low-dimensional embedding is centred at the origin. As mentioned above, this maximisation can be reformulated as the following semidefinite programming problem.

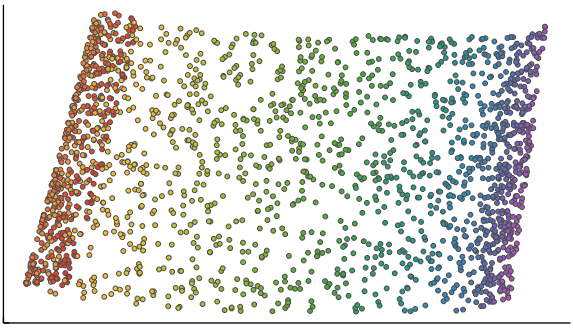

Fig. 2.5 Example 2-dimensional embeddings of the S-Curve dataset found by Maximum Variance Unfolding with $k = 13$

Maximise $trace(\mathbf{F})$ subject to:

(1) $\mathbf{F} \preceq 0$.

(2) $\sum_{ij} \mathbf{f}_{ij} = 0$.

(3) $\mathbf{k}_{ii} - 2\mathbf{k}_{ij} + \mathbf{k}_{jj} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \; \forall \; E(i, j) \in G$.

Constraint (1) ensures that the matrix is positive semidefinite and Constraint (3) ensures local isometry [10].

MVU differs from other spectral techniques in that rather than constructing a feature matrix from measurable properties (i.e. covariance, Euclidean distance), it directly learns the feature matrix by solving a convex optimisation problem. Once the feature matrix has been learnt however, MVU fits in with other spectral techniques as the low-dimensional embedding is given as the top eigenvectors of Eq. (2.1).

The low-dimensional embedding of the S-Curve dataset found using MVU is shown in Fig. 2.5.

2.3.3 Diffusion Maps

The Diffusion Maps framework [11] has its roots in the field of dynamical systems. Whereas Isomap measures interpoint distances as geodesics, and MVU measures them as 'un-rolled' Euclidean distances, Diffusion Maps uses the idea of diffusion distance to capture the relationships between data points. Diffusion Maps works on a fully connected model of the data, so unlike Isomap, which considers single shortest paths, Diffusion Maps considers several paths through the data making it potentially more robust to noise. Figure 2.6 shows the 2-dimensional embedding of the S-Curve dataset founding using Diffusion Maps.

The feature matrix, $\mathbf{F}$, found by Diffusion Maps contains the diffusion distances between data points after t time steps. The diffusion distance can be found by computing a Markov random walk on a weighted graph $G = \langle V, E \rangle$, with the vertex set

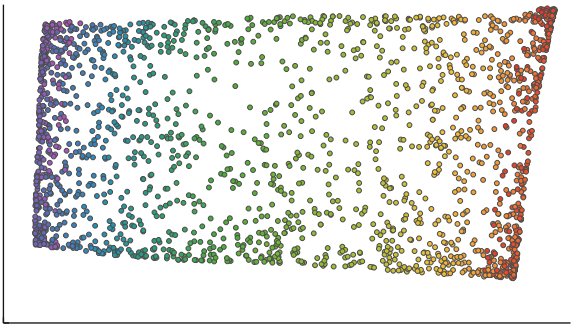

Fig. 2.6 Example 2-dimensional embeddings of the S-Curve dataset found by Diffusion Maps with $t = 1, \sigma = 0.1$

corresponding to $\mathbf{X}$ and the edge set built using the Gaussian kernel such that

$$E(i, j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}} \tag{2.6}$$

where σ is the variance of the Gaussian. This edge set can be recast in terms of an affinity matrix such that $\mathbf{W}_{ij} = E(i, j)$, and is subsequently row normalised (the rows of $\mathbf{W}$ sum to 1) to adjust for the influence of local geometry versus distribution across the manifold [11]. The initial transition matrix $\mathbf{P}^{(1)}$ is then formed with entries $\mathbf{p}_{ij}^{(1)} = \frac{\mathbf{w}_{ij}}{d(\mathbf{x}_i)}$, where $d(\mathbf{x}_i)$ is the degree of node $\mathbf{x}_i$ such that $d(\mathbf{x}_i) = \sum_j \mathbf{w}_{ik}$. Each entry, $\mathbf{p}_{ij}^{(1)}$, can be interpreted as the forward transition probability of a diffusion process between $\mathbf{x}_i$ and $\mathbf{x}_j$ in a single timestep. The transition matrix $\mathbf{P}^{(1)}$ therefore reflects the first-order neighbourhood structure of G. Following on from this, the transition matrix $\mathbf{P}^{(t)}$ represents the probability of the diffusion process after t timesteps, and $\mathbf{p}_{ij}^{(t)}$ corresponds to the probability of going from $\mathbf{x}_i$ to $\mathbf{x}_j$ in t timesteps.

Using the forward probabilities from the random walk at timestep t, the diffusion distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ can be defined by

$$\mathbf{D}^{(t)}(\mathbf{x}_i, \mathbf{x}_j) = \sum_k \frac{(\mathbf{p}_{ik}^{(t)} - \mathbf{p}_{jk}^{(t)})^2}{\phi_0(\mathbf{x}_k)} \tag{2.7}$$

where $\phi_0(\mathbf{x}_k)$ is the unique stationary distribution defined as $\frac{d(\mathbf{x}_i)}{\sum_j d(\mathbf{x}_j)}$. This term is used to assign more weight to denser regions of the graph. As can be seen from Eq. (2.7), the diffusion distance is low between data points with a high forward transition probability.

As shown in [11], the low-dimensional representation is found by the decomposition of a modified form of Eq. (2.1) such that

$$\mathbf{P}^{(t)}\mathbf{Q} = \Lambda\mathbf{Q} \tag{2.8}$$

The largest eigenvalue is trivial since the graph represented by $\mathbf{P}^{(t)}$ is fully connected, and so the associated eigenvector, $\mathbf{u}_1$, is discarded. The eigenvectors are then scaled by their eigenvalues leading to the d-dimensional embedding $\mathbf{Y} = \{\Lambda_2\mathbf{u}_2, \Lambda_3\mathbf{u}_3, \ldots, \Lambda_{d+1}\mathbf{u}_{d+1}\}$.

2.3.4 Locally Linear Embedding (LLE)

Although published at the same time as Isomap, LLE [12] presents a very different approach to spectral dimensionality reduction. Whereas Isomap, MVU, and Diffusion Maps, seek to construct a dense distance matrix to model global geodesic distances, LLE constructs a sparse feature matrix based on the local linear structure of the manifold. As such, LLE derives the low-dimensional embedding from the bottom eigenvectors of a sparse feature matrix and is considered a *local* technique for spectral dimensionality reduction. At a broad level, LLE works by initially describing each input data point in terms of the weights needed to reconstruct it from its nearest neighbours. These weights correspond to a description of the local geometry of each data point, and as such, the low-dimensional embedding of the data can be found by reconstructing each data point in the low-dimensional space in terms of the previously found weights. LLE is therefore a two step framework, with the weights being found in the first step by "locking" the data points, and then finding the location of the low-dimensional data points by "locking" the weights found in the first step.

The simple intuition behind LLE is that each input point, $\mathbf{x}_i$, and its k-nearest neighbours are locally linear, that is, they lie on or near a linear 'patch' of the manifold. By making this assumption the local geometry can be characterised by linear reconstruction weights, $\mathbf{W}$, that reconstruct each point $\mathbf{x}_i$ from its k-nearest neighbours. The weights are measured by the squared distance cost function

$$\varepsilon(\mathbf{W}) = \sum_i \left\| \mathbf{x}_i - \sum_j \mathbf{W}_{ij}\mathbf{x}_j \right\|^2 \tag{2.9}$$

which is minimised with the constraints: (i) $\mathbf{W}_{ij} = 0$ if $\mathbf{x}_j$ is not in the k neighbourhood of $\mathbf{x}_i$; (ii) $\sum_j \mathbf{W}_{ij} = 1$ for all i. This sparse weight matrix therefore describes the local geometry of the data up to size k. Once the weights in $\mathbf{W}$ have been found by solving the least squares problem of Eq. (2.9) using the technique described in [12], LLE seeks to reconstruct the data in the low-dimensional space by minimising the embedding cost function

$$\Phi(\mathbf{Y}) = \sum_i \left\| \mathbf{y}_i - \sum_j \mathbf{W}_{ij}\mathbf{y}_j \right\|^2 \tag{2.10}$$

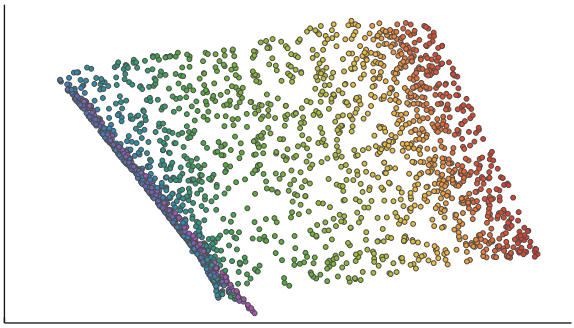

Fig. 2.7 Example 2-dimensional embeddings of the S-Curve dataset found by Locally Linear Embedding with $k = 8$

with the specific constraints: (i) $\sum_i \mathbf{y}_i = 0$, that is, the outputs are centered; (ii) the outputs have unit covariance. These constraints prevent a degenerate solution and also allow the minimisation of Eq. (2.10) to be found through the computation of the bottom $(d + 1)$ eigenvectors of the feature matrix $\mathbf{F} = (\mathbf{I} - \mathbf{W})^T(\mathbf{I} - \mathbf{W})$, where the bottom eigenvector (corresponding to the constant unit vector) is discarded.

Figure 2.7 shows the 2-dimensional embedding of the S-Curve dataset found using LLE. As can be seen, since LLE is a local approach to dimensionality reduction, the global shape of the manifold is not fully recovered.

Although similar methods existed before LLE to model the local geometry of a manifold (e.g. Local PCA [13]), the novel feature of LLE is that it maps the local geometric models into a single, coherent, coordinate system. As well as this, by modelling the local geometry in terms of reconstruction coefficients, the core methodology of LLE can be used to embed new data points into a previously learnt manifold, a problem which is described in more detail in Chap. 5.

2.3.5 Laplacian Eigenmaps

Similar to LLE, Laplacian Eigenmaps [14] seeks to find a low-dimensional embedding through the preservation of local neighbourhood properties. As in LLE, a weight matrix is employed to capture the local structure of the data, and the low-dimensional embedding is found through the bottom $(d + 1)$ eigenvectors of this sparse feature matrix. While LLE appeals to geometric intuition to model the local properties of the data, Laplacian Eigenmaps appeals to spectral graph theory and the notion of the graph Laplacian to find a solution to the minimisation of the central cost function.

Laplacian Eigenmaps seeks to minimise the following embedding cost function:

$$\Phi(\mathbf{Y}) = \sum_{ij} (\mathbf{y}_i - \mathbf{y}_j)^2 \mathbf{w}_{ij} \tag{2.11}$$

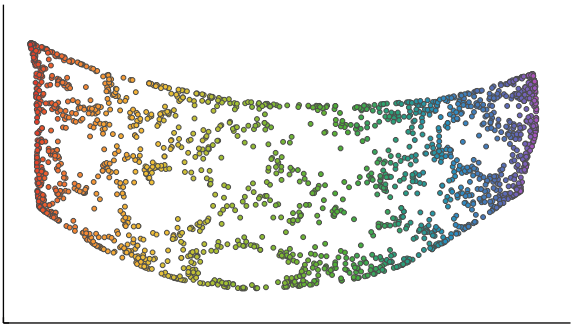

Fig. 2.8 Example 2-dimensional embeddings of the S-Curve dataset found by Laplacian Eigenmaps with $k = 12, \sigma = 2$

where large weights in $\mathbf{W}$ correspond to small distances in the high-dimensional space and so nearby points in $\mathbf{X}$ are brought together in $\mathbf{Y}$ with 'nearness' being represented by the weight values in $\mathbf{W}$. The weight matrix can be constructed in numerous ways with the most common forms being either to assign constant weights, $\mathbf{w}_{ij} = 1/k$, or to use a heat kernel to exponentially decay the weights, $\mathbf{w}_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2)/\sigma^2$ where σ is a scale parameter defining the size of the kernel.

As shown in [14], the solution to Eq. (2.11) can be found by recasting it in terms of a general eigenproblem involving the Laplacian of the graph. The Laplacian matrix, $\mathbf{F}$, is defined as $\mathbf{F} = \mathbf{M} - \mathbf{W}$ where $\mathbf{M}$ is the degree matrix of $\mathbf{W}$ such that $\mathbf{m}_{ij} = \sum_j \mathbf{w}_{ij}$. With this in mind, the cost function of Eq. (2.11) can be re-formulated as

$$\begin{aligned} \Phi(\mathbf{Y}) &= \sum_{ij} (\mathbf{y}_i - \mathbf{y}_j)^2 \mathbf{w}_{ij} \\ &= \sum_{ij} (\mathbf{y}_i^2 + \mathbf{y}_j^2 - 2\mathbf{y}_i \mathbf{y}_j) \mathbf{w}_{ij} \\ &= \sum_i \mathbf{y}_i^2 \mathbf{D}_{ii} + \sum_j \mathbf{y}_j^2 \mathbf{D}_{jj} - 2 \sum_{ij} \mathbf{y}_i \mathbf{y}_j \mathbf{w}_{ij} \\ &= 2\mathbf{Y}^T \mathbf{F} \mathbf{Y} \end{aligned} \tag{2.12}$$

Therefore, the minimisation of Eq. (2.11) can be found by observing that $\mathbf{F}$ is positive semidefinite, and so the values of $\mathbf{Y}$ that minimise the objective function are given by the solution to the eigenvalue problem $\mathbf{F}\mathbf{Y} = \Lambda\mathbf{D}\mathbf{Y}$. As with LLE, the bottom (smallest) eigenvector is discarded and the $(d+1)$ smallest eigenvectors give rise to the q-dimensional embedding. Figure 2.8 shows the low-dimensional embedding of the S-Curve dataset found using Laplacian Eigenmaps.

One further point of interest with regards to Laplacian Eigenmaps is the algorithm's linearised variant, Locality Preserving Projections (LPP) [15]. LPP seeks to compute a transformation matrix using the graph Laplacian that maps the data points into a subspace. Specifically, LPP seeks to minimise

$$\underset{\substack{\mathbf{a} \\ \mathbf{a}^T\mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{a}=1}}{\arg\min} \quad \mathbf{a}^T\mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{a} \tag{2.13}$$

where $\mathbf{a}$ is a transformation vector. The vectors, $\mathbf{a}$, that minimise the above objective function are given by the smallest eigenvectors of the generalised eigenproblem $\mathbf{X}\mathbf{F}\mathbf{X}^T\mathbf{a} = \lambda\mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{a}$. Locality Preserving Projections has proved useful in many problem domains due to its linearity and speed, and also due to the fact that it naturally incorporates the mapping of new points. This is discussed in more detail in Chap. 5.

2.3.6 Local Tangent Space Alignment (LTSA)

The Local Tangent Space Alignment method [16] models the data in terms of the local tangent space of each data point. The tangent space for a data point can conceptually be thought of as an approximation of the principal manifold at a local scale passing through the data point. The tangent spaces for a set of near neighbours are assumed to be overlapping, and so the global co-ordinate system can be found by aligning the local tangent spaces. LTSA therefore builds a global representation of the data by aligning a set of local models.

The local information for a data point $\mathbf{x}_i$ is calculated by finding the q largest eignvectors of the correction matrix $\mathbf{W}_i$ of the neighbourhood around $\mathbf{x}_i$ such that

$$\mathbf{W}_i = (\mathbf{X}_{\mathscr{N}_i} - \bar{\mathbf{x}}_i\mathbf{e}^T)^T(\mathbf{X}_{\mathscr{N}_i} - \bar{\mathbf{x}}_i\mathbf{e}^T) \tag{2.14}$$

where $\mathbf{X}_{\mathscr{N}_i}$ corresponds to the points in $\mathbf{X}$ that appear in the k-neighbourhood of $\mathbf{x}_i$ represented as $\mathscr{N}_i$ and $\mathbf{e}$ is a k-dimensional column vector of ones. The local information, $\mathbf{G}_i$, is then built from the largest eignvectors of $\mathbf{W}_i$ such that $\mathbf{G}_i = [\mathbf{e}/\sqrt{k}, \mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_q]$ where $\mathbf{e}/\sqrt{k}$ is a centering term.

LTSA seeks to minimise a cost function that minimises the distances between points in the low-dimensional space and the tangent space. As shown in [16], the solution to this minimisation problem is formed by the d smallest eigenvectors of an alignment matrix $\mathbf{F}$. The alignment matrix is found by iteratively summing over all local information matrices:

$$\mathbf{F}_{\mathscr{N}_i,\mathscr{N}_i} \leftarrow \mathbf{F}_{\mathscr{N}_i,\mathscr{N}_i} + \mathbf{H}_k(\mathbf{I} - \mathbf{G}_i\mathbf{G}_i^T)\mathbf{H}_k \tag{2.15}$$

starting from $\mathbf{F} = 0$ for $\forall ij$. Here, $\mathbf{H}_k$ is a centering matrix of size k with $\mathbf{H}_k = \mathbf{I}_k - \frac{1}{k}\mathbf{e}\mathbf{e}^T$ where e is a k-dimensional column vector of all ones.

The low-dimensional representation $\mathbf{Y}$ is given by the bottom $(d+1)$ eigenvectors of $\mathbf{F}$ found according to Eq. (2.1) with the smallest eigenvector being discarded. An example low-dimensional embedding of the S-Curve dataset found using Local Tangent Space Alignment is given in Fig. 2.9.

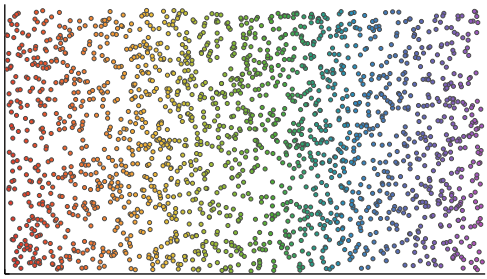

Fig. 2.9 Example 2-dimensional embeddings of the S-Curve dataset found by Local Tangent Space Alignment with $k = 10$

Table 2.1 The kernel form of each technique is described relative to each specific feature matrix $\mathbf{F}$

Method	Properties	Kernel form
Linear [1, 4]	Top eigenvectors	$\mathbf{K} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ where κ is an appropriately chosen kernel function
Isomap [2]	Top eigenvectors	$\mathbf{K} = -\frac{1}{2}\mathbf{H}\tau(\mathbf{S})\mathbf{H}$ where $\tau(\mathbf{S})$ is the squared distance matrix and $\mathbf{H}$ is the centering matrix
MVU [8]	Top eigenvectors	As given (Sect. 2.3.2)
Diffusion maps [11]	Top eigenvectors	As given (Sect. 2.3.3)
LLE [12]	Bottom eigenvectors	$\mathbf{K} = \mathbf{H}(\lambda_{\max}\mathbf{I} - \mathbf{F})\mathbf{H}$ where $\lambda_{\max}$ is the maximum eigenvalue of $\mathbf{F}$
Eigenmaps [14]	Bottom eigenvectors	$\mathbf{K} = \mathbf{H}\mathbf{F}^{\dagger}\mathbf{H}$ where $\mathbf{F}^{\dagger}$ is the pseudo-inverse of the Laplacian matrix
LTSA [16]	Bottom eigenvectors	$\mathbf{K} = \mathbf{I} - \mathbf{F}$

For a definition of $\mathbf{F}$ with respect to each technique see Sects. 2.2.1–2.3.6

2.4 Kernel Formulation

There have been numerous attempts to phrase spectral dimensionality reduction within a unified framework, examples being Graph Embedding [17], and a unified probabilistic framework [18]. But perhaps one of the most significant and useful unified settings is found when phrasing spectral dimensionality reduction algorithms as kernel methods [19]. All spectral methods can be described as performing Kernel Principal Components Analysis [20] on specially constructed Gram matrices (e.g. Kernel Isomap, LLE, and Laplacian Eigenmaps [19], Kernel LTSA [21]). It is worth noting that Maximum Variance Unfolding [8] and Diffusion Maps [11] already construct kernel matrices so they do not need to be adapted to fit within this framework. Table 2.1 summarises the kernel form of each of the discussed spectral dimensionality reduction algorithms. These kernel forms will be useful in later chapters when solutions from the kernel methods community can be applied to kernel based spectral dimensionality reduction techniques.

2.5 Summary

Spectral dimensionality reduction seeks to obtain a low-dimensional embedding of a high-dimensional dataset through the eigendecomposition of a specially constructed feature matrix. This feature matrix will capture certain properties of the data such as inter-point covariance or local linear reconstruction weights. The different methods of formulating this feature matrix will have different implications for various open problems, as will be seen in later chapters.

References

1. Joliffe, I.T.: Principal Component Analysis. Springer-Verlag, New York (1986)
2. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2322 (2000)
3. Hotelling, H.: Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology **24**, 417–441 (1933)
4. Cox, T.F., Cox, M.A.A.: Multidimensional Scaling. Chapman and Hall (2001)
5. Saul, L.K., Weinberger, K.Q., Ham, J.H., Sha, F., Lee, D.D.: Semisupervised Learning, chap. Spectral Methods for Dimensionality Reduction, pp. 293–308. MIT Press, Cambridge, MA (2006)
6. Rowlinson, P.: Graph Connections: Relationships Between Graph Theory and other Areas of Mathematics, chap. Linear Algebra, pp. 86–99. Oxford Science Publications (1997)
7. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1**, 269–271 (1959)
8. Weinberger, K.Q., Packer, B.D., Saul, L.K.: Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In: In Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, pp. 381–388 (2005)
9. Vandenberghe, L., Boyd, S.P.: Semidefinite programming. SIAM Review **38**(1), 49–95 (1996)
10. Weinberger, K.Q., Sha, F., Saul, L.K.: Learning a kernel matrix for nonlinear dimensionality reduction. In: In Proceedings of the 21st International Conference on Machine Learning, pp. 839–846 (2004)
11. Lafon, S., Lee, A.B.: Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(9), 1393–1403 (2006)
12. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by Locally Linear Embedding. Science **290**, 2323–2326 (2000)
13. Kambhatla, N., Leen, T.K.: Dimension reduction by local Principal Components Analysis. Neural Computation **9**(7), 1493–1516 (1994)
14. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in Neural Information Processing Systems 14: Proceedings of the 2002 Conference (NIPS), pp. 585–591 (2002)
15. He, X., Niyogi, P.: Locality Preserving Projections. In: Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference (NIPS), pp. 153–160. MIT Press (2003)
16. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimension reduction via local tangent space alignment. SIAM Journal on Scientific Computing **26**(1), 313–338 (2004)
17. Yan, S., Xu, D., Zhang, B., Zhang, H.J., Yang, Q., Lin, S.: Graph embedding: A general framework for dimensionality reduction. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(1), 40–51 (2007)

18. Lawrence, N.D.: A unifying probabilistic perspective for spectral dimensionality reduction: Insights and new models. Journal of Machine Learning Research **13**, 1609–1638 (2012)
19. Ham, J., Lee, D.D., Mika, S., Schölkopf, B.: A kernel view of the dimensionality reduction of manifolds. In: In Proceedings of the 21st International Conference on Machine Learning, pp. 47–55 (2004)
20. Scholkopf, B., Smola, E., Bottou, L., Burges, C., Bultho, H., Gegenfurtner, K., Ner, P.H.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation **10**, 1299–1319 (1998)
21. Ma, L., Crawford, M.M., Tian, J.W.: Generalised supervised local tangent space alignment for hyperspectral image classification. Electronics Letters **46**(7), 497–498 (2010)

http://www.springer.com/978-3-319-03942-8

Open Problems in Spectral Dimensionality Reduction

Strange, H.; Zwiggelaar, R.

2014, IX, 92 p. 20 illus., 15 illus. in color., Softcover
ISBN: 978-3-319-03942-8