

# Chapter 1

## Graph-based dimensionality reduction

JOHN A. LEE

Center for Molecular Imaging, Radiotherapy, and Oncology (MIRO)  
Université catholique de Louvain  
Brussels, Belgium  
john.lee@uclouvain.be

MICHEL VERLEYSSEN

Machine Learning Group (MLG), ICTEAM Institute  
Université catholique de Louvain  
Louvain-la-Neuve, Belgium  
Statistique, Analyse, Modélisation Multidisciplinaire (SAMM)  
Université Paris I Panthéon-Sorbonne  
Paris, France  
michel.verleysen@uclouvain.be

---

### 1.1 Summary

Dimensionality reduction aims at representing high-dimensional data into low-dimensionality space. In order to make sense, the low-dimensional representation, or embedding, has to preserve some well-defined structural properties of data. The general idea is that similar data items should be displayed close to each other, whereas longer distances should separate dissimilar ones. This principle applies to data that can

be either sampled from a manifold of the data space or distributed among several clusters. In both cases, one generally agrees that rendering the local structure comes prior to reproducing the global arrangement of data. In practice, the structural properties can be pairwise measurements such as dissimilarities (all kinds of distances) or similarities (decreasing functions of distances). Relative proximities such as rank of sorted distances can be used as well. Within this context, graphs can serve several purposes. For instance, they can model the fact that (dis)similarities are missing or overlooked. Even if all pairwise dissimilarities are available, one can consider that only the shortest ones are meaningful and reflect the local data structure. Graphs corresponding to  $K$ -ary neighborhoods or  $\epsilon$ -balls can be built and utilized to compute the length of shortest paths or commute-time distances associated to random walks.

This chapter reviews some of the most prominent dimensionality reduction methods that rely on graphs. These include several techniques based on geodesic distances, such as Isomap and its variants. Spectral methods involving a graph Laplacian are described as well. More biologically inspired techniques, such as the self-organizing map, identify a topographic mapping between a predefined graph and the data manifold. Illustrative experiments focus on image data (scanned handwritten digits) and results are assessed using rank-based quality criteria.

## 1.2 Introduction

The interpretation of high-dimensional data remains a difficult task, mainly because human vision is not used to deal with spaces having more than three dimensions. Part of this difficulty stems from the curse of dimensionality, a convenient expression that encompasses all weird and unexpected properties of high-dimensional spaces. Dimensionality reduction (DR) aims at constructing an alternative low-dimensional representation of data, in order to improve readability and interpretability. Of course, this low-dimensional representation must be meaningful and faithful to the genuine data. In practice, the representation must preserve important structural properties of the data set, such as relative proximities, similarities or dissimilarities. The general idea is that dissimilar data items should be represented far from each other, whereas similar ones should appear close to each other. Dimensionality reduction serves other purposes than just data visualization. For instance, DR can be used in data compression and denoising. Dimensionality reduction can also preprocess data, with the hope that a simplified representation can accelerate any subsequent processing or improve its outcome.

Linear DR is well known, with techniques such as principal component analysis [1] (PCA) and classical metric multidimensional scaling [2, 3] (MDS). The former tries to preserve the covariances in the low-dimensional space, whereas the latter attempts to reproduce the Gram matrix of pairwise dot products. Nonlinear dimensionality reduction [4] (NLDR) emerged later, with nonlinear variants of multidimensional

scaling [5, 6, 7] such as Sammon’s nonlinear mapping [8] (NLM) and curvilinear component analysis [9, 10] (CCA). Most of these methods are based on the preservation of pairwise distances. Research in NLDR is multidisciplinary and follows many approaches, ranging from artificial neural networks [11, 12, 13, 9, 14] to spectral techniques [15, 16, 17, 18, 19, 20, 21]. If linear DR assumes that data are distributed within or near a linear subspace, NLDR necessitates more complex models.

The most generic framework consists in assuming that data are sampled from a smooth manifold. For this reason, modern NLDR is sometimes referred to as manifold learning [22, 20]. Under this hypothesis, one seeks to re-embed the manifold in a space having the lowest possible dimensionality, without modifying its topological properties. In practice, smooth manifolds are difficult to conciliate with the discrete nature of data. In contrast, graph structures have proven to be very useful and tight connections between NLDR and graph embedding [23, 24, 25] exist.

Another usual hypothesis is to assume that data are sampled from clusters rather than from a manifold. Dimensionality reduction methods that emphasize clusters are often closely related to spectral clustering [26, 27, 28, 29]. In this domain, graphs are very handy as well, with useful tools such as the Laplacian matrix.

This chapter is organized as follows. Section 1.3 introduces the necessary notations and classical methods such as PCA and MDS. Section 1.4 details how graphs can help introducing nonlinearities in the DR methods. Next, Sections 1.5 and 1.6 review some of the major DR methods that perform manifold learning by using either distances (‘global’ methods) or similarities (‘local’ methods). Section 1.7 deals with graph embedding, another paradigm used in nonlinear dimensionality reduction. Finally, Section 1.8 compares the methods on a few examples, before Section 1.9 that draws the conclusions and sketches some perspectives for the near future.

### 1.3 Classical methods

Let  $\mathbf{X} = [\mathbf{x}_i]_{1 \leq i \leq N}$  denotes a multivariate data set, with  $\mathbf{x}_i \in \mathbb{R}^M$ . Let  $\mathbf{Y} = [\mathbf{y}_i]_{1 \leq i \leq N}$  denote its low-dimensional representation, with  $\mathbf{y}_i \in \mathbb{R}^P$  and  $P < M$ . Dimensionality reduction aims at finding a transformation  $\mathcal{T}$  from  $\mathbb{R}^{M \times N}$  to  $\mathbb{R}^{P \times N}$  that minimizes the reconstruction error defined by

$$E = \|\mathbf{X} - \mathcal{T}^{-1}(\mathbf{Y})\|_2^2, \quad (1.1)$$

where  $\mathbf{Y} = \mathcal{T}(\mathbf{X})$  and  $\|\mathbf{U}\|_2^2 = \text{Tr}(\mathbf{U}^T \mathbf{U}) = \text{Tr}(\mathbf{U} \mathbf{U}^T)$  is the Frobenius norm of  $\mathbf{U}$ . In other words, the encoding-decoding process resulting from the successive application of  $\mathcal{T}$  and  $\mathcal{T}^{-1}$  should produce minimal distortion.

### 1.3.1 Principal component analysis

The simplest option for  $\mathcal{T}$  is obviously a linear transformation. In principal component analysis [30, 31, 32, 33, 1], this transformation can be written as  $\mathcal{T}(\mathbf{X}) = \mathbf{V}_P^T(\mathbf{X} - \mathbf{u}\mathbf{1}^T)$ , where  $\mathbf{u}$  is an offset and  $\mathbf{V}_P$  is an orthogonal matrix ( $\mathbf{V}_P^T\mathbf{V}_P = \mathbf{I}$ ) with  $P$  columns. Orthogonality reduces the number of free parameters in  $\mathbf{V}_P$  and provides the  $P$ -dimensional subspace of  $\mathbb{R}^M$  with a basis of orthonormal vectors. With such a linear transformation, the reconstruction error can be rewritten as

$$E(\mathbf{u}, \mathbf{V}_P; \mathbf{X}) = \|(\mathbf{X} - \mathbf{u}\mathbf{1}^T) - \mathbf{V}_P\mathbf{V}_P^T(\mathbf{X} - \mathbf{u}\mathbf{1}^T)\|_2^2 \quad (1.2)$$

$$= \|(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)(\mathbf{X} - \mathbf{u}\mathbf{1}^T)\|_2^2 \quad (1.3)$$

$$= \text{Tr}((\mathbf{X} - \mathbf{u}\mathbf{1}^T)^T(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)(\mathbf{X} - \mathbf{u}\mathbf{1}^T)) \quad (1.4)$$

$$= \text{Tr}((\mathbf{X} - \mathbf{u}\mathbf{1}^T)^T(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)(\mathbf{X} - \mathbf{u}\mathbf{1}^T)) \quad (1.5)$$

In order to identify  $\mathbf{u}$ , we can compute the partial derivative of the reconstruction error with respect to  $\mathbf{u}$  and equate it with zero. Defining  $\mathbf{E} = \mathbf{X} - \mathbf{u}\mathbf{1}^T$ , we have

$$\frac{\partial E(\mathbf{u}, \mathbf{V}_P; \mathbf{X})}{\partial \mathbf{u}} = \frac{\partial \text{Tr}(\mathbf{E}^T(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)\mathbf{E})}{\partial \mathbf{E}} \frac{\partial \mathbf{E}^T}{\partial \mathbf{u}} = 2(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)\mathbf{E}\mathbf{1} = \mathbf{0} \quad (1.6)$$

Assuming that  $\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T$  has full rank, we have  $(\mathbf{X} - \mathbf{u}\mathbf{1}^T)\mathbf{1}^T = \mathbf{0}$  and therefore  $\mathbf{u} = \mathbf{X}\mathbf{1}^T/N$ . The optimal offset is thus the sample mean of  $\mathbf{X}$ . Knowing this, we can simplify the transformation into  $\mathcal{T}(\mathbf{X}) = \mathbf{V}_P^T\mathbf{X}$  if we assume that  $\mathbf{X}\mathbf{1} = \mathbf{0}$ . The reconstruction error is then rewritten as

$$E(\mathbf{V}_P; \mathbf{X}) = \|(\mathbf{I} - \mathbf{V}_P\mathbf{V}_P^T)\mathbf{X}\|_2^2 = \text{Tr}(\mathbf{X}\mathbf{X}^T) - \text{Tr}(\mathbf{V}_P^T\mathbf{X}\mathbf{X}^T\mathbf{V}_P) \quad (1.7)$$

where the first term is constant. The second term can be minimized under the constraint  $\mathbf{V}_P^T\mathbf{V}_P = \mathbf{I}$  using Lagrange's technique. The Lagrangian can be written as

$$L(\mathbf{V}_P, \mathbf{\Lambda}; \mathbf{X}) = \text{Tr}(\mathbf{V}_P^T\mathbf{X}\mathbf{X}^T\mathbf{V}_P) + \text{Tr}(\mathbf{\Lambda}(\mathbf{I} - \mathbf{V}_P^T\mathbf{V}_P)) \quad (1.8)$$

where  $\mathbf{\Lambda}$  is the diagonal matrix containing Lagrange's multipliers. The partial derivative of  $L(\mathbf{V}_P, \mathbf{\Lambda}; \mathbf{X})$  with respect to  $\mathbf{V}_P$  is given by

$$\frac{\partial L(\mathbf{V}_P, \mathbf{\Lambda}; \mathbf{X})}{\partial \mathbf{V}_P} = 2\mathbf{X}\mathbf{X}^T\mathbf{V}_P - 2\mathbf{\Lambda}\mathbf{V}_P \quad (1.9)$$

After equating the partial derivative with zero and rearranging the terms, we obtain  $\mathbf{\Lambda}\mathbf{V}_P = \mathbf{X}\mathbf{X}^T\mathbf{V}_P$ , which turns out to be an eigenproblem. Without loss of generality and because  $\mathbf{X}$  is centered, the product  $\mathbf{X}\mathbf{X}^T$  can be replaced with the sample covariance  $\mathbf{C}(\mathbf{X}) = \mathbf{X}\mathbf{X}^T/N$ . The covariance matrix is symmetric and positive semidefinite by construction. Its eigenvalues are therefore larger than or equal to zero. If we write the eigenvalue decomposition as  $\mathbf{C}(\mathbf{X}) = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ , then the solution of the maximization problem for a given  $P$

is provided by the eigenvectors in  $\mathbf{V}$  that are associated with the  $P$  largest eigenvalues. These eigenvectors give the columns of  $\mathbf{V}_P = [\mathbf{v}_i]_{1 \leq i \leq P}$ . The covariance in the  $P$ -dimensional subspace is given by

$$\mathbf{C}(\mathbf{Y}) = \mathbf{Y}\mathbf{Y}^T / N = \mathbf{V}_P^T \mathbf{X}\mathbf{X}^T \mathbf{V}_P / N = \mathbf{V}_P^T \mathbf{C}(\mathbf{X}) \mathbf{V}_P = \mathbf{V}_P^T \mathbf{V}_P \mathbf{\Lambda} \mathbf{V}_P^T \mathbf{V}_P = \mathbf{\Lambda}_P, \quad (1.10)$$

where  $\mathbf{\Lambda}_P$  denotes the restriction of  $\mathbf{\Lambda}$  to its first  $P$  rows and columns. As the covariance is diagonal in the subspace, this shows that PCA also decorrelates the data set. Matrix  $\mathbf{V}_P$  is also the solution that minimizes  $\|\mathbf{C}(\mathbf{X}) - \mathbf{V}_P \mathbf{\Lambda}_P \mathbf{V}_P^T\|_2^2$ . This shows that minimal reconstruction error is equivalent to variance preservation. If  $\mathbf{V}_P$  maximizes  $\text{Tr}(\mathbf{V}_P^T \mathbf{X}\mathbf{X}^T \mathbf{V}_P)$ , then it also trivially maximizes  $\text{Tr}(\mathbf{X}^T \mathbf{V}_P \mathbf{V}_P^T \mathbf{X})$ .

### 1.3.2 Multidimensional scaling

In the previous section, PCA requires the data set to be available in the form of coordinates in  $\mathbf{X}$ . In contrast, classical metric multidimensional scaling [2, 3](CMMDS) starts from the Gram matrix, which is defined as  $\mathbf{G}(\mathbf{X}) = \mathbf{X}^T \mathbf{X}$  and is positive semidefinite by construction. The aim and model of CMMDS are basically the same as those of PCA. The solution remains the same as well, but it is computed differently. To see this, let us use the singular value decomposition of  $\mathbf{X}$ . It can be written as  $\mathbf{X} = \mathbf{V}\mathbf{S}\mathbf{U}^T$ , where both  $\mathbf{U}$  and  $\mathbf{V}$  are orthogonal matrices and  $\mathbf{S}$  contains the singular values on its diagonal in descending order of magnitude. The covariance matrix can be rewritten as  $\mathbf{C}(\mathbf{X}) = \mathbf{X}\mathbf{X}^T / N = \mathbf{V}\mathbf{S}\mathbf{U}^T \mathbf{U}\mathbf{S}^T \mathbf{V}^T / N = \mathbf{V}(\mathbf{S}\mathbf{S}^T / N)\mathbf{V}^T$ . The last expression is equivalent to the eigenvalue decomposition of the covariance matrix if we rename the product  $\mathbf{S}\mathbf{S}^T / N$  into  $\mathbf{\Lambda}$ . Similarly, the Gram matrix can be rewritten as  $\mathbf{G}(\mathbf{X}) = \mathbf{X}^T \mathbf{X} = \mathbf{U}\mathbf{S}^T \mathbf{V}^T \mathbf{V}\mathbf{S}\mathbf{U}^T = \mathbf{U}(\mathbf{S}^T \mathbf{S})\mathbf{U}^T$ . The last expression is equivalent to the eigenvalue decomposition of the Gram matrix and shows the tight connection with the eigenvalue decomposition of the covariance matrix. The coordinates in the  $P$ -dimensional subspace can be rewritten as  $\mathbf{Y} = \mathbf{V}_P^T \mathbf{X} = \mathbf{V}_P \mathbf{V}\mathbf{S}\mathbf{U}^T = \mathbf{S}_P \mathbf{U}^T$ , where  $\mathbf{S}_P$  is the restriction of  $\mathbf{S}$  to its  $P$  first columns. This shows that the coordinates are given by the  $P$  leading eigenvectors of the Gram matrix, scaled by the square root of their associated eigenvalue.

Until here, we have assumed that  $\mathbf{X}$  is centered in the Gram matrix. If it is not, then we need to compute

$$\mathbf{G}(\mathbf{X}) = (\mathbf{X} - (\mathbf{X}\mathbf{1}/N)\mathbf{1}^T)^T (\mathbf{X} - (\mathbf{X}\mathbf{1}/N)\mathbf{1}^T) = (\mathbf{I} - \mathbf{1}\mathbf{1}^T / N)(\mathbf{X}^T \mathbf{X})(\mathbf{I} - \mathbf{1}\mathbf{1}^T / N). \quad (1.11)$$

This shows that the Gram matrix can be centered as such, without explicitly knowing  $\mathbf{X}$ . The centering matrix  $\mathbf{I} - \mathbf{1}\mathbf{1}^T / N$  is a powerful tool that can also be used if the data set consists of pairwise squared Euclidean distances. In this case, we write

$$\mathbf{\Delta}^2(\mathbf{X}) = [\|\mathbf{x}_i - \mathbf{x}_j\|_2^2]_{1 \leq i, j \leq N} = \text{diag}(\mathbf{G}(\mathbf{X}))\mathbf{1}^T - 2\mathbf{G}(\mathbf{X}) + \mathbf{1}\text{diag}(\mathbf{G}(\mathbf{X}))^T, \quad (1.12)$$

where operator  $\text{diag}$  provides the column vector built from the diagonal entries of its argument. Double centering, namely, multiplying the matrix of squared distances by the centering matrix on both the left and

right sides, yields

$$(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)\mathbf{\Delta}^2(\mathbf{X})(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N) = -2(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)\mathbf{G}(\mathbf{X})(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N) . \quad (1.13)$$

This results from

$$\text{diag}(\mathbf{G}(\mathbf{X}))\mathbf{1}^T(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N) = \text{diag}(\mathbf{G}(\mathbf{X}))\mathbf{1}^T - \text{diag}(\mathbf{G}(\mathbf{X}))\mathbf{1}^T\mathbf{1}\mathbf{1}^T/N \quad (1.14)$$

$$= \text{diag}(\mathbf{G}(\mathbf{X}))\mathbf{1}^T - \text{diag}(\mathbf{G}(\mathbf{X}))\mathbf{1}^T \quad (1.15)$$

$$= \mathbf{0}\mathbf{0}^T \quad (1.16)$$

and similarly  $(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)\mathbf{1}\text{diag}(\mathbf{G}(\mathbf{X}))^T = \mathbf{0}\mathbf{0}^T$ . Classical metric MDS is thus a flexible method that can be applied to coordinates, pairwise dot products, or pairwise squared Euclidean distances. For a given  $P$ , it can be shown that  $\mathbf{Y} = \mathbf{S}_P\mathbf{U}^T$  minimizes a cost function called STRAIN [34] and defined as

$$E(\mathbf{Y}; \mathbf{X}) = \|\mathbf{G}(\mathbf{X}) - \mathbf{G}(\mathbf{Y})\|_2^2 = \sum_{i,j} (\mathbf{x}_i^T \mathbf{x}_j - \mathbf{y}_i^T \mathbf{y}_j)^2 . \quad (1.17)$$

In other words, CMMDS tries to preserve dot products in the linear subspace.

### 1.3.3 Nonlinear MDS and distance preservation

Given the close relationship between dot products and squared Euclidean distances, one might extend the CMMDS principle to distance preservation. Unfortunately, distance preservation cannot be achieved with spectral decomposition such as in PCA and CMMDS. It requires more generic optimization tools such as gradient descent or ad hoc algorithms. As an advantage, these tools are quite flexible and allow for more freedom in the definition of the cost function to be minimized. This freedom also means that the  $P$ -dimensional coordinates are no longer constrained to a linear transformation of those in the  $M$ -dimensional space. For instance, one can consider the STRESS function [6] defined as  $E(\mathbf{D}; \mathbf{\Delta}) = \|\mathbf{\Delta}(\mathbf{X}) - \mathbf{\Delta}(\mathbf{Y})\|_2^2$ . If  $\delta_{ij}(\mathbf{X})$  and  $\delta_{ij}(\mathbf{Y})$  denote the pairwise distance in the  $M$ - and  $P$ -dimensional space respectively, then a more general form of the STRESS is given by

$$E(\mathbf{Y}; \mathbf{X}) = \sum_{i,j} w_{ij} (\delta_{ij}(\mathbf{X}) - \delta_{ij}(\mathbf{Y}))^2 , \quad (1.18)$$

where  $w_{ij}$  controls the weight of each difference of distances in the cost function. These weights can be chosen arbitrarily or can depend on  $\delta_{ij}$  and/or  $d_{ij}$ . The emphasis put on the preservation of small distances can be reinforced by defining  $w_{ij} = 1/\delta_{ij}$ , such as in Sammon's stress function, which is used in his nonlinear mapping (NLM) technique [8]. Another possibility is to define  $w_{ij} = f(\delta_{ij})$ , where  $f$  is a non-increasing positive function. For instance, in curvilinear component analysis (CCA) [10], we have  $w_{ij} = H(\lambda_i - \delta_{ij})$ , where  $H$  is a step function and  $\lambda_i$  a width parameter.

Yet another cost function for MDS is the squared stress, often shortened as SSTRESS [7], whose generic definition is

$$E(\mathbf{Y}; \mathbf{X}) = \sum_{i,j} w_{ij} (\delta_{ij}^2(\mathbf{X}) - \delta_{ij}^2(\mathbf{Y}))^2 . \quad (1.19)$$

As distances are squared, the SSTRESS is actually closer to the STRAIN of CMMDS.

Various optimization procedures can be used to minimize these cost functions. Gradient descent with a diagonal approximation of the Hessian is used in Sammon's NLM, whereas CCA relies on a stochastic gradient descent. Yet another technique consists of successive majorizations of the STRESS with convex functions whose minimum can be computed analytically [34].

## 1.4 Nonlinearity through graphs

The projection of data onto linear subspaces, such as in the methods described in the previous section, might be insufficient in many cases. As an emblematic example, let us consider a manifold looking like the thin layer of jam in a Swiss roll cake (see Section 1.8 and Fig. 1.1 for an illustration). This data set is distributed in a three-dimensional space on a surface shaped as a spiral. As a matter of fact, the underlying manifold is a two-dimensional rectangular subspace, which actually corresponds to a latent parameterization of the Swiss roll manifold. No linear projection is able to provide a satisfying representation of this latent space: there is always the risk that pieces of the Swiss roll will be superimposed in the representation. Intuitively, the solution to this problem would be to first unroll the manifold and to perform the projection afterwards. All NLDR methods implement this intuition in some way. They explore the idea sketched at the end of the previous section: local neighborhood relationships should be faithfully rendered in the low-dimensional representation, whereas more distant relationships are less important. Unrolling the Swiss roll puts this principle into practice: close points remain near each other, while the distances between non-neighbors is increased. Graphs provide an efficient way to encode neighborhood relationships. For instance, graphs can represent  $K$ -ary neighborhoods (the set of the  $K$  nearest neighbors around each data point) or  $\varepsilon$ -ball neighborhoods (the set of all neighbors lying within a ball of radius  $\varepsilon$  centered on each data point). However, graphs also generate sparsity, which has to be compensated for. The variety of NLDR methods reflects the different possibilities to intelligently fill the holes left by sparsity.

Three categories of methods can be distinguished:

- Some methods start with pairwise distances, keep those associated with local neighborhoods only, and replace the missing distances with values that facilitate manifold unfolding. Eventually, CMMDS or any of its nonlinear variants can be used to compute the low-dimensional representation. Typical methods in this category are Isomap [16] (and all other methods using graph distances [35, 36, 37, 38,

39)) and maximum variance unfolding [20].

- Some methods define pairwise similarities (or affinities, proximities, vicinities, i.e. any positive quantity that decreases with distance). Non-neighboring data items are generally given a null similarity, which leads to a sparse similarity matrix. Next, the sparse similarity matrix is converted into a distance matrix that can be processed by CMMDS. Typical methods in this category are Laplacian eigenmaps [40, 18] and locally linear embedding [17, 22].
- Some methods rely on what could be called graph placement or graph embedding [24, 25]. These are often ad hoc methods that are driven by mechanical concepts such as spring forces applied to masses. A graph can be used to represent the masses and the springs that connect them. Two subcategories exist. The graph can be defined a priori in the low-dimensional space and placed in the high-dimensional one or the graph can be built in a data-dependent way in the high-dimensional and the placement achieved in the low-dimensional one. The self-organizing map [11, 41] belongs to the first subcategory whereas force-directed placement, Isotop [42], and the exploratory observation machine (XOM) [43] are in the second one.

## 1.5 Graph-based distances

Let us imagine a manifold made of a curved sheet of paper, which is embedded in our physical three-dimensional world. The Swiss roll is an example of such manifold. Let us also consider the Euclidean distance between two relatively distant points of the manifold. Depending on the sheet curvature, the distance will vary, although the sheet keeps the same size (paper does not allow any stretching or shrinking). This shows that distance preservation makes little sense if the manifold to be embedded in a lower-dimensional space needs some unfolding or unrolling. This issue arises because Euclidean distances measure lengths along straight lines whereas the manifold occupies a nonlinear subspace. The solution to this problem is obviously to compute distances "as the ant crawls" instead of "as the crow flies". In other words, distances should be computed along the manifold or, more accurately, along manifold geodesic curves. In a smooth manifold, the geodesic curve between two points on the manifold is the smooth one-dimensional submanifold with the shortest length. The term geodesic distance refers to this length. In a manifold such as a sheet of paper, geodesic distances are invariant to curvature changes. Therefore, geodesic distances capture the internal structure of the manifold without influence from the way it is embedded.

As a matter of fact, geodesic distances cannot be evaluated if no analytical expression of the manifold is available. However, if at least some points of the manifold are known, for instance through data set  $\mathbf{X}$ , then geodesic distances can be approximated by using a graph [44]. Each data vector  $\mathbf{x}_i$  is associated with

a graph vertex and either  $K$ -ary neighborhoods or  $\varepsilon$ -ball neighborhoods provide the edges. Such a graph yields a discrete representation of the manifold. Within this framework, the geodesic distance between  $\mathbf{x}_i$  and  $\mathbf{x}_j$  can be approximated by the length of the shortest path that connects the two corresponding vertices. Shortest paths and their lengths can be easily computed by Dijkstra's or Floyd's algorithms [45, 46].

At this point, we know that geodesic distances approximated by graph distances can characterize the internal structure of a manifold. But how can we force its unfolding in view of dimensionality reduction? The solution consists in trying to reproduce the graph distances measured in the high-dimensional space with Euclidean distances in the low-dimensional space. In this way, geodesic curves are matched with straight lines. In practice, several methods can perform this hybrid distance preservation. For instance, CMMDS can be applied to the matrix containing all squared pairwise shortest path lengths (instead of squared Euclidean distances). This method is known as Isomap [16]. Although CMMDS is purely linear, Isomap achieves a nonlinear embedding: the computation of the shortest path lengths can be thought of as being equivalent to applying a nonlinear transformation to the data. Nonlinear variants of CMMDS can be used as well, such as Sammon's nonlinear mapping [8] or curvilinear component analysis [10], for instance. This yields geodesic Sammon mapping [37, 38] and curvilinear distance analysis [35, 36].

Graph distances have three main drawbacks. First, let us consider a smooth manifold  $\mathcal{M}$  that is isometric to some subset of a Euclidean space. Let us assume that  $\Delta_{\mathcal{M}}^2(\mathbf{X})$  contains the actual squared geodesic distances for some manifold points stored in  $\mathbf{X}$ . If we compute  $-1/2(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)\Delta_{\mathcal{M}}^2(\mathbf{X})(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)$ , then we should end up with a valid Gram matrix, which is positive semidefinite. Unfortunately, this statement does not hold true if geodesic distances are not exact. This means that if  $\Delta_{\mathcal{G}}^2(\mathbf{X})$  contains shortest path lengths for some graph  $\mathcal{G}$  instead of geodesic distances, the product  $-1/2(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)\Delta_{\mathcal{G}}^2(\mathbf{X})(\mathbf{I} - \mathbf{1}\mathbf{1}^T/N)$  will not necessarily be positive semidefinite. The spectral decomposition used in Isomap can thus yield negative eigenvalues. In practice, their magnitude is often negligible and workarounds are described in the literature [16, 47, 48]. This issue matters only for methods relying on spectral decomposition, while it is negligible for methods that rely on a nonlinear variant of CMMDS such as Sammon's NLM or CCA.

A second drawback arises with manifolds that are not isometric to some subset of a Euclidean space. A piece of a spherical surface, for instance, is not isometric to a piece of plane. In this case, distance preservation can only be imperfect. The weighting schemes used in the nonlinear variant of CMMDS can partly address this issue. Isometry is also lost in the case of a non-convex manifold. For example, a sheet of paper with a hole is not isometric to a piece of plane: some geodesic distances are forced to circumvent the hole in the manifold. Again, a weighted distance preservation giving more importance to short distances can help.

The third shortcoming of graph distances is related to the graph construction. If the value of the parameters  $K$  or  $\varepsilon$  is not appropriately chosen, there might be too few or too many edges in the graph. Too

few edges typically lead to an overestimation of the actual geodesic distances (paths are zigzagging). In some cases, the graph representation of the manifold can comprise disconnected parts, yielding to infinite distances. Too many edges can cause significant underestimation of some geodesic distances. This could happen if a "short circuit" edge accidentally connects two points that are not actual neighbors.

If we compare graph distances to the corresponding Euclidean ones, ones sees that the former are longer than or equal to the latter. We have seen that the use of these longer distances in CMMDS can be seen as a way to unfold the manifold. However, Dijkstra's and Floyd's algorithms compute shortest paths in a greedy way, without taking into account the objective of DR. This means that they build a matrix of pairwise distances but there is no guarantee that this matrix is optimal for DR. For instance, in the case of Isomap (CMMDS with graph distances), there is no explicit optimization of the Gram matrix such that the eigenspectrum energy concentrates into a minimal number of eigenvalues.

This issue [48] is addressed in maximum variance unfolding (MVU) [20]. MVU starts with a neighborhood graph ( $K$  nearest neighbors or  $\varepsilon$ -balls). Just as in Isomap, the idea to achieve manifold unfolding is to stretch distances between non-neighboring data points. For this purpose, one considers a matrix of squared pairwise distances  $\Delta^2(\mathbf{Y}) = [\delta_{ij}^2(\mathbf{Y})]_{1 \leq i, j \leq N}$  and the simple objective function to be maximized:

$$E(\mathbf{Y}) = \frac{1}{2N} \mathbf{1}^T \Delta(\mathbf{Y}) \mathbf{1} = \frac{1}{2N} \sum_{i,j} \delta_{ij}^2(\mathbf{Y}) \quad , \quad (1.20)$$

subject to  $\delta_{ij}^2(\mathbf{Y}) = \delta_{ij}^2(\mathbf{X}) = \|\mathbf{x}_i - \mathbf{x}_j\|_2^2$  if  $\mathbf{x}_i \sim \mathbf{x}_j$ . Additional constraints stem from the assumption that  $\Delta(\mathbf{Y})$  contains the pairwise Euclidean distances for some  $P$ -dimensional representation  $\mathbf{Y}$  of data set  $\mathbf{X}$ . In other words,  $\Delta^2(\mathbf{Y})$  must satisfy the equality  $\Delta^2(\mathbf{Y}) = \text{diag}(\mathbf{G}(\mathbf{Y})) \mathbf{1}^T - 2\mathbf{G}(\mathbf{Y}) + \mathbf{1} \text{diag}(\mathbf{G}(\mathbf{Y}))^T$ , where  $\mathbf{G}\mathbf{Y} = \mathbf{X}^T \mathbf{X}$  is the Gram matrix for some  $\mathbf{Y}$ . Without loss of generality, one can assume that the mean of  $\mathbf{Y}$  is null, i.e.  $\mathbf{X}\mathbf{1} = \mathbf{0}$  and  $\mathbf{1}^T \mathbf{G}(\mathbf{Y}) \mathbf{1} = \mathbf{1}^T \mathbf{Y}^T \mathbf{Y} \mathbf{1} = 0$ . This helps to reformulate the problem with only the Gram matrix. Indeed we can write

$$E(\mathbf{Y}) = \frac{1}{2N} \mathbf{1}^T \Delta^2(\mathbf{Y}) \mathbf{1} \quad (1.21)$$

$$= \frac{1}{2N} \mathbf{1}^T (\text{diag}(\mathbf{G}(\mathbf{Y})) \mathbf{1}^T - 2\mathbf{G}(\mathbf{Y}) + \mathbf{1} \text{diag}(\mathbf{G}(\mathbf{Y}))^T) \mathbf{1} \quad (1.22)$$

$$= \frac{1}{2N} \mathbf{1}^T (\text{diag}(\mathbf{G}(\mathbf{Y})) \mathbf{1}^T + \mathbf{1} \text{diag}(\mathbf{G}(\mathbf{Y}))^T) \mathbf{1} = \text{Tr}(\mathbf{G}(\mathbf{Y})) \quad . \quad (1.23)$$

In order for CMMDS to be applicable to matrix  $\mathbf{G}(\mathbf{Y})$ , the following constraints must be satisfied:

- To be a Gram matrix,  $\mathbf{G}(\mathbf{Y})$  must be positive semidefinite.
- If  $\mathbf{G}(\mathbf{Y})$  factorizes into centered coordinates, then the product  $\mathbf{1}^T \mathbf{G}(\mathbf{Y}) \mathbf{1}$  is equal to 0.
- For all neighbors  $\mathbf{x}_i \sim \mathbf{x}_j$ , we must have  $g_{ii}(\mathbf{Y}) - 2g_{ij}(\mathbf{Y}) + g_{jj}(\mathbf{Y}) = \|\mathbf{x}_i - \mathbf{x}_j\|_2^2$ .

Provided the neighborhood graph is fully connected, the last set of constraints also controls the scale of the embedding  $\mathbf{Y}$ . Such a constrained optimization problem involving a positive semidefinite matrix can be solved using semidefinite programming [49]. Once the optimal  $\mathbf{G}(\mathbf{Y})$  is determined, an eigenvalue decomposition such as in CMMDS can factorize it in order to identify  $\mathbf{Y}$ .

Both Isomap and MVU can be decomposed in a two-step optimization procedure, where the second step is CMMDS applied to a modified distance matrix. The difference resides in the first step of each method. In Isomap, Dijkstra's or Floyd's algorithm minimizes the length of vertex-to-vertex paths. While this objective proves to be useful to unfold a manifold, it is not directly related to DR. In contrast, MVU implements a similar idea (distances must be stretched) in a more principled way, with an objective function that takes into account the needs and constraints of the subsequent CMMDS step.

## 1.6 Graph-based similarities

The previous section has shown that graphs provide a handy framework to build dissimilarities that are more relevant than the Euclidean distances when it comes to nonlinear manifold unfolding. However, graphs can also formalize similarities, which are usually defined as decreasing positive functions of the corresponding distances. Similarities provide therefore a very natural way to emphasize the local structure of data, such as neighborhoods around each data point. In contrast, the global structure and large distances are associated with small similarity values. If the latter are neglected, the matrix of pairwise similarities becomes sparse and a graph can efficiently represent it.

### 1.6.1 Laplacian eigenmaps

In Laplacian eigenmaps [40, 18], the idea to achieve manifold unfolding is the dual of that of MVU. Instead of stretching the distances between non-neighboring data points, the distances between neighbors are shrunk in the embedding space. For this purpose, the edges of the neighborhood graph are annotated with similarities; non-neighboring points have null similarities. The simplest similarity definition is

$$w_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \sim \mathbf{x}_j \\ 0 & \text{otherwise} \end{cases} . \quad (1.24)$$

As the neighborhood graph is undirected, matrix  $\mathbf{W} = [w_{ij}]_{1 \leq i, j \leq N}$  is symmetric in addition to be sparse.

The minimization of the distances between neighbors can be achieved with the cost function

$$E(\mathbf{Y}; \mathbf{W}) = \frac{1}{2} \sum_{i,j} w_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2 \quad (1.25)$$

$$= \frac{1}{2} \sum_{i,j} w_{ij} (\mathbf{y}_i^T \mathbf{y}_i + \mathbf{y}_j^T \mathbf{y}_j - 2\mathbf{y}_i^T \mathbf{y}_j) \quad (1.26)$$

$$= \sum_i \mathbf{y}_i^T \left( \sum_j w_{ij} \right) \mathbf{y}_i - \sum_{i,j} \mathbf{y}_i^T w_{ij} \mathbf{y}_j \quad (1.27)$$

$$= \text{Tr}(\mathbf{Y}^T \mathbf{D} \mathbf{Y}) - \text{Tr}(\mathbf{Y}^T \mathbf{W} \mathbf{Y}) = \text{Tr}(\mathbf{Y} \mathbf{L} \mathbf{Y}^T) , \quad (1.28)$$

where  $\mathbf{D}$  is a diagonal matrix with  $d_{ii} = \sum_j w_{ji} = \sum_j w_{ij}$  and  $\mathbf{L} = \mathbf{D} - \mathbf{W}$  is the unnormalized Laplacian matrix of the graph whose edges are weighted with  $\mathbf{W}$ .

The minimization of  $E(\mathbf{Y}; \mathbf{W})$  admits the trivial solution  $\mathbf{Y} = \mathbf{00}^T$ . In order to avoid this, we impose the scale constraint  $\mathbf{Y} \mathbf{D} \mathbf{Y}^T = \mathbf{I}$ . This leads to the Lagrangian

$$L(\mathbf{Y}, \mathbf{\Lambda}; \mathbf{W}) = \text{Tr}(\mathbf{Y} \mathbf{L} \mathbf{Y}^T) - \text{Tr}(\mathbf{\Lambda} (\mathbf{Y} \mathbf{D} \mathbf{Y}^T - \mathbf{I})) = \text{Tr}(\mathbf{Y} \mathbf{L} \mathbf{Y}^T) - \text{Tr}(\mathbf{Y}^T \mathbf{\Lambda} \mathbf{D} \mathbf{Y}) + \text{Tr}(\mathbf{\Lambda}) , \quad (1.29)$$

The partial derivative with respect to  $\mathbf{Y}$  is

$$\frac{\partial L(\mathbf{Y}, \mathbf{\Lambda}; \mathbf{W})}{\partial \mathbf{Y}} = 2\mathbf{Y} \mathbf{L} - 2\mathbf{Y} \mathbf{\Lambda} \mathbf{D} . \quad (1.30)$$

Equating the partial derivative to zero and rearranging the terms leads to  $\mathbf{L} \mathbf{Y}^T = \mathbf{\Lambda} \mathbf{D} \mathbf{Y}^T$ , which is a generalized eigenvalue problem. As  $\mathbf{D}$  is diagonal, it is equivalent to  $\tilde{\mathbf{L}} \mathbf{Y}^T = \mathbf{\Lambda} \mathbf{Y}^T$ , where  $\tilde{\mathbf{L}} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$  is the normalized Laplacian. Both the unnormalized and normalized Laplacian matrices are positive semidefinite. Notice that they are also singular since by construction  $\mathbf{L} \mathbf{1} = \mathbf{D} \mathbf{1} - \mathbf{W} \mathbf{1} = \mathbf{0}$ . The multiplicity of the zero eigenvalue is given by the number of connected components in the graph. As we look for a value of  $\mathbf{Y}$  with  $P$  rows that minimizes  $E(\mathbf{Y}; \mathbf{W})$ , the solution is provided by the trailing  $P$  eigenvectors of  $\mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2}$ , namely, those associated with the eigenvalues having the lowest nonzero magnitude. Eventually, if  $\tilde{\mathbf{L}} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^T$ , then  $\mathbf{X} = \mathbf{U}_P^T$ , where  $\mathbf{U}_P$  denotes the restriction of  $\mathbf{U}$  to its first  $P$  columns.

The embedding found by Laplacian eigenmaps corresponds to a multivariate extension of the solution to a min-cut problem in a graph [50]. There are several variants to Laplacian eigenmaps, depending on the way the graph edges are weighted. Moreover, the unnormalized Laplacian can replace the normalized one. This amounts to changing the scale constraint to  $\mathbf{Y} \mathbf{Y}^T = \mathbf{I}$ . Let us also consider yet another scaling possibility, namely  $\mathbf{Y} = \mathbf{\Omega}_P^{1/2} \mathbf{U}_P^T$ , where  $\mathbf{\Omega}_P$  is a diagonal matrix composed of the inverse of the nonzero smallest  $P$  eigenvalues  $\lambda_i^{-1}$  in descending order. The corresponding Gram matrix is given by  $\mathbf{U}_P \mathbf{\Omega}_P \mathbf{U}_P^T$ . If we assume that the graph consists of a single connected component, then the zero eigenvalue has multiplicity one and  $\tilde{\mathbf{L}}^+ = \mathbf{U}_{N-1} \mathbf{\Omega}_{N-1} \mathbf{U}_{N-1}^T$  is the Moore-Penrose pseudo-inverse of  $\tilde{\mathbf{L}}$ . This means that Laplacian

eigenmaps is equivalent to CMMDS applied to a Gram matrix that is the pseudo-inverse of the normalized Laplacian. The computation of the Laplacian and its inversion are equivalent to a nonlinear transformation applied to the data set  $\mathbf{X}$  and denoted by  $\mathbf{Z} = \phi(\mathbf{X})$ . The Gram matrix in this space is given by  $\mathbf{G}(\mathbf{Z}) = \tilde{\mathbf{L}}^+$  and the corresponding Euclidean distances are  $\mathbf{1} \text{diag}(\mathbf{G}(\mathbf{Z}))^T - 2\mathbf{G}(\mathbf{Z}) + \text{diag}(\mathbf{G}(\mathbf{Z}))\mathbf{1}^T$ . These distances are referred to as commute time distances [27, 51]. They are closely related to diffusion distances [28] and also to the length (or duration) of random walks in a neighborhood graph whose edges are weighted with transition probabilities. An analogy with electrical networks is possible as well [52], with graph edges being resistances and the commute time distances being the global effective resistance between two given vertices through the whole network. This analogy allows establishing a formal relationship between distances (resistances) and similarities that are inversely proportional to distances (conductances); it also highlights the duality between DR methods relying on a dense matrix of distances and those involving a sparse matrix of similarities.

From a more general point of view, the notation  $\mathbf{G}(\mathbf{Z})$  used above with  $\mathbf{Z} = \phi(\mathbf{X})$  translates the idea that CMMDS can be applied to nonlinearly transformed coordinates. Most of the time, this transformation remains implicit. For instance, commute time distances in Laplacian eigenmaps or geodesic distances in Isomap induce an (hopefully useful) transformation that promotes CMMDS from a linear DR method to a nonlinear one, while keeping many advantages, such as a convex optimization. Maximum variance unfolding goes a step further and actually customizes transformation  $\phi$ . Along with locally linear embedding described in the next section, all these spectral methods can be cast within the framework of kernel PCA [15]. Kernel PCA is the ancestor of all spectral methods and relies on key properties of Mercer's kernels. Such a kernel is a smooth function  $\kappa(\mathbf{x}_i, \mathbf{x}_j)$  that is symmetric with respect to its arguments and induces a scalar product in a so-called feature space  $\mathcal{F}$ . In other words, we have  $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{F}}$  where  $\phi: \mathbb{R}^M \rightarrow \mathcal{F}, \mathbf{x} \mapsto \mathbf{z} = \phi(\mathbf{x})$ . Taken the other way round, this property allows  $\phi$  to be induced from a given Mercer kernel  $\kappa$ . In particular, one can build a matrix  $[\kappa(\mathbf{x}_i, \mathbf{x}_j)]_{1 \leq i, j \leq N}$  with the guarantee that it is a Gram matrix in some feature space  $\mathcal{F}$ . Sample mean removal in  $\mathcal{F}$  can be performed indirectly as well, just by pre- and postmultiplying the Gram matrix by centering matrix  $\mathbf{I} - \mathbf{1}\mathbf{1}^T/N$ . Somewhat of a misnomer, kernel PCA actually applies CMMDS to this 'kernelized' Gram matrix in order to identify  $\mathbf{Z}$  and subsequently a linear  $P$ -dimensional projection  $\mathbf{Y}$  of  $\mathbf{Z}$  with maximum variance. The pioneering work about kernel PCA [15] establishes the theoretical framework of spectral NLDR but provides no clue as to which Mercer kernel performs the best in practice.

## 1.6.2 Locally linear embedding

The idea behind locally linear embedding [17] (LLE) is to represent each data point as a regularized linear mixture of its neighbors. For points sampled from a smooth manifold, the resulting linear coefficients can be assumed to depend only upon local proximity relationships. Topological operations applied to the manifold, such as unfolding and flattening, have thus a minor impact on these coefficients. Therefore, the same coefficients could be reused to determine a new data embedding in a lower-dimensional space.

In practice, LLE relies on the availability of a neighborhood graph, built with  $K$ -ary neighborhoods or  $\varepsilon$ -balls. The first step of LLE is to identify the reconstruction coefficients in the high-dimensional data space. For this purpose, LLE uses a first cost function defined as

$$E(\mathbf{W}; \mathbf{X}) = \frac{1}{2} \sum_i \left\| \mathbf{x}_i - \sum_j w_{ij} \mathbf{x}_j \right\|_2^2 = \frac{1}{2} \sum_i \left\| \sum_j w_{ij} (\mathbf{x}_i - \mathbf{x}_j) \right\|_2^2, \quad (1.31)$$

where  $\mathbf{W}$  is subject to the following constraints:  $\mathbf{W}\mathbf{1} = \mathbf{1}$ ,  $w_{ii} = 0$ , and  $w_{ij} = 0$  if and only if  $\mathbf{x}_i$  and  $\mathbf{x}_j$  are not neighbors. Each row of  $\mathbf{W}$  can be identified separately. Let  $\mathbf{G}_i$  denote the local Gram-like matrix involving all neighbors of  $\mathbf{x}_i$ . It can be written as  $\mathbf{G}_i = [(\mathbf{x}_k - \mathbf{x}_i)^T (\mathbf{x}_l - \mathbf{x}_i)]_{k,l}$ , where  $\mathbf{x}_k \sim \mathbf{x}_i$  and  $\mathbf{x}_l \sim \mathbf{x}_i$ . If vector  $\mathbf{w}_i$  contains the nonzero entries of the  $i$ th row of  $\mathbf{W}$ , then we have  $\mathbf{w}_i = \min_{\mathbf{w}} \mathbf{w}^T \mathbf{G}_i \mathbf{w}$  with  $\mathbf{w}^T \mathbf{1} = 1$ . The solution is given by

$$\mathbf{w}_i = \frac{\mathbf{G}_i \mathbf{1}}{\mathbf{1}^T \mathbf{G}_i \mathbf{1}}. \quad (1.32)$$

In order to avoid trivial solutions when the rank of  $\mathbf{G}_i$  is lower than the number of neighbors  $K$ , it is advised in [17, 22, 53] to replace  $\mathbf{G}_i$  with  $\mathbf{G}_i + (\Delta^2 \text{Tr}/K)(\mathbf{G}_i) \mathbf{I}$  with  $\Delta = 0.1$ . This regularization scheme prevents trivial solutions, such as  $w_{ij} = 0$  if  $\mathbf{x}_i \sim \mathbf{x}_j$ . To some extent,  $\mathbf{W}$  can be interpreted as a sparse similarity matrix.

Once the reconstruction coefficients are known, a second cost function can be defined, where  $\mathbf{W}$  is fixed and coordinates are the unknown:

$$E(\mathbf{Y}; \mathbf{W}) = \sum_i \left\| \mathbf{y}_i - \sum_j w_{ij} \mathbf{y}_j \right\|_2^2 \quad (1.33)$$

$$= \sum_i \mathbf{y}_i^T \mathbf{y}_i - 2 \sum_i \mathbf{y}_i^T \left( \sum_j w_{ij} \mathbf{y}_j \right) + \sum_{i,j} w_{ij} \mathbf{y}_i^T \mathbf{y}_j w_{ij} \quad (1.34)$$

$$= \text{Tr}(\mathbf{Y}^T \mathbf{Y}) - 2 \text{Tr}(\mathbf{Y}^T \mathbf{Y} \mathbf{W}^T) + \text{Tr}(\mathbf{W} \mathbf{Y}^T \mathbf{Y} \mathbf{W}^T) \quad (1.35)$$

$$= \text{Tr}(\mathbf{Y} \mathbf{I} \mathbf{Y}^T) - 2 \text{Tr}(\mathbf{Y} \mathbf{I} \mathbf{W} \mathbf{Y}^T) + \text{Tr}(\mathbf{Y} \mathbf{W}^T \mathbf{W} \mathbf{Y}^T) = \text{Tr}(\mathbf{Y} (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W}) \mathbf{Y}^T). \quad (1.36)$$

The last equality shows that LLE is similar to Laplacian eigenmaps. The product  $\mathbf{M} = (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W}) = \mathbf{I} - (\mathbf{W}^T + \mathbf{W} - \mathbf{W}^T \mathbf{W})$  is symmetric and positive semidefinite by construction. This product has the same structure as a normalized Laplacian matrix, with the diagonal elements of the first term being equal to the

sum of the rows (or columns) of the second term. For the first term, we have  $\text{diag} \mathbf{I} = \mathbf{1}$ , whereas the second term leads to  $(\mathbf{W}^T + \mathbf{W} - \mathbf{W}^T \mathbf{W}) \mathbf{1} = \mathbf{W}^T \mathbf{1} + \mathbf{1} - \mathbf{W}^T \mathbf{1}$ , knowing that  $\mathbf{W} \mathbf{1} = \mathbf{1}$ . With respect to  $\mathbf{W}$ , which is generally not symmetric, the product in  $\mathbf{M}$  can be seen as a kind of squared Laplacian.

The minimization of LLE's second objective function can be achieved with a spectral decomposition of  $\mathbf{M} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^T$ . The matrix  $\mathbf{M}$  is singular, as  $\mathbf{M} \mathbf{1} = (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W}) \mathbf{1} = \mathbf{0} = \mathbf{0} \mathbf{1}$ . This shows that  $\mathbf{1}$  is an eigenvector of  $\mathbf{M}$  with zero as associated eigenvalue. As in Laplacian eigenmaps, the multiplicity of the null eigenvalue is given by the number of connected components in the neighborhood graph. The solution of the minimization is formed by the transpose of the trailing  $P$  eigenvectors in  $\mathbf{U}$ , namely, those associated with the smallest strictly positive eigenvalues in  $\mathbf{\Lambda}$ .

As with Laplacian eigenmaps, the solution can be rescaled in various ways. For instance, one might consider  $\mathbf{Y} = \mathbf{\Lambda}_P^{-1/2} \mathbf{U}^T$ , where  $\mathbf{\Lambda}_P$  is a diagonal matrix made of the smallest nonzero  $P$  eigenvalues of  $\mathbf{\Lambda}$  in ascending order. This particular solution can be cast within the framework of CMMDS and corresponds to a Gram matrix equal to the Moore-Penrose pseudo-inverse of  $\mathbf{M}$ .

## 1.7 Graph embedding

This section deals with more heuristic ways to use graphs in dimensionality reduction. The described methods are of two kinds. In the first category, a predefined graph with a fixed planar representation is fitted in the high-dimensional data space. The self-organizing maps are the most widely known example of this kind. The second category works in the opposite direction. A graph is build in the high-dimensional space, according to the data distribution; next this graph is embedded in a low-dimensional visualization space.

### 1.7.1 From LD to HD: self-organizing maps

A self-organizing map (SOM) [41] can be seen as a nonlinear generalization of PCA. As detailed in Section 1.3.1, the objective of PCA is to find a linear subspace that minimizes the reconstruction error. Let us assume that this linear subspace is a two-dimensional plane. Intuitively, PCA places this plane amidst the cloud of data points, by minimizing the distances between the points and their projection on the plane. In order to extend PCA to nonlinear subspaces, one could replace the plane by some manifold. The SOM implements this idea by substituting the continuous plane with a discretized representation such as an articulated grid.

Let us assume that a grid is defined by an undirected graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ . Each vertex  $v_i$  is equipped with coordinates  $\xi_k$  in the  $M$ -dimensional data space and  $\gamma_k$  in the  $P$ -dimensional grid space. Each edge  $e_{kl}$  is weighted with a positive number  $w_{kl}$  that indicates the distance between  $v_k$  and  $v_l$ . If no edge connects  $v_k$  and  $v_l$ , then the distance is infinite. In order to obtain a simple and readable visualization of data, coordinates  $\gamma_k$  are often chosen in such a way that the grid nodes are regularly placed within a rectangle. Direct neighbors

of  $v_k$  are typically located on a square or a hexagon (honeycomb configuration).

In PCA, the objective function measures the distortion between the genuine data points and their projection on the linear subspace. In a SOM, data points are projected onto the closest grid node. Hence, several data points can be associated with the same grid node, which plays a similar role as a centroid in a vector quantization [54] technique or in  $K$ -means-like algorithms [55]. For each data point  $\mathbf{x}_i$ , the index of the closest grid node is given by  $\ell = \arg \min_k \|\mathbf{x}_i - \xi_k\|_2$ . Just as in a  $K$ -means algorithm, the SOM tries to minimize the distortion between the data and the grid nodes. In the SOM, however, an additional objective must be reached concurrently: grid neighbors must remain as close as possible in the data space. These two concurrent objectives are difficult to formalize into a simple cost function, although some attempts exist [56]. For this reason, an iterative and heuristic procedure is typically used to update the coordinates in  $\Xi = [\xi_k]_{1 \leq k \leq Q}$ . The simplest procedure is inspired by biological considerations: the SOM works like a neural network that progressively learns a set of patterns. In practice, the data set  $\mathbf{X}$  contains the patterns, which are each presented several times to the SOM in random order. Let us assume that  $\mathbf{x}_i$  is considered. The first step is to determine the closest grid node  $\xi_\ell$  as described above. Next, all grid nodes will be updated according to

$$\xi_k \leftarrow \xi_k + \alpha f(w_{k\ell}/\sigma)(\mathbf{x}_i - \xi_k) \quad , \quad (1.37)$$

where  $\alpha$  is a learning rate (or step size),  $f$  is a positive decreasing function of its argument, and  $\sigma$  is a sort of bandwidth (or neighborhood radius). The learning rate can be decreased either after each presentation of a data vector or only between two complete sweeps of the whole data set. Function  $f$  can be a flipped step function or a decreasing exponential. Parameter  $\sigma$  controls the grid elasticity. If  $\sigma$  is large,  $f(w_{k\ell}/\sigma)$  is maximal for  $k = \ell$  and will not significantly decrease for close neighbors, which will thus closely follow the movement of  $\xi_\ell$ . On the contrary, if  $\sigma$  is low,  $\xi_\ell$  has less influence on its neighbors. If the SOM is to be compared to an assembly of springs and masses, then the lower  $\sigma$  is, the heavier the masses are and the weaker the springs are.

Self-organizing maps are still widely used in various fields such as exploratory data analysis and data visualization. There are many variants of the basic algorithm described above and many ways to display [57] and assess its outcome [58, 59, 60, 61, 62]. There also probabilistic variants such as the generative topographic mapping [63], which relies on a Bayesian approach and maximizes a log-likelihood by an iterative expectation-maximization procedure. The neural gas algorithm [64, 65] works in a similar way as the SOM, although it has no predefined graph structure; hence the neural gas does not provide a low-dimensional representation.

## 1.7.2 From HD to LD: Isotop

An SOM, as it is described in the previous section, has several shortcomings. First, it relies on centroids (the grid nodes) like a  $K$ -means algorithm. This means that the SOM does not really provide a visualization of each data point. Second, the SOM embeds a predefined graph in the data space, whereas one usually expects a DR method to work the other way round by embedding the data set in a low-dimensional space. There are actually many methods that address those two limitations. Most of these methods are loosely related to the fields of graph embedding [23] and graph drawing [24, 25, 66]. Their general principle is first to build a neighborhood graph in the high-dimensional data space and next to embed this graph in a lower-dimensional visualization space using graph layout algorithms. Most of these algorithms are heuristic and find their inspiration in mechanics. For instance, in a force-based layout, the graph vertices are masses and edges are springs connecting them. The final layout or embedding then results from a free energy minimization that equilibrates the mass-spring system. As an illustrative example, we describe hereafter Isotop [67, 42, 4], a method that is very close to a SOM.

Let us assume that data set  $\mathbf{X}$  is a sample drawn from some unknown distribution in a high-dimensional space. In order to obtain a lower-dimensional representation  $\mathbf{Y}$  of  $\mathbf{X}$ , let us further assume that the support of this distribution is a manifold that can be mapped into a low-dimensional space. Starting from this pair of distributions, we can consider a new point with coordinates  $\mathbf{x}$  and  $\mathbf{y}$  in the high- and low-dimensional spaces, respectively. If we want this point to have the same neighbors in both spaces, we can define the cost function of Isotop as

$$E(\mathbf{Y}; \mathbf{X}, \rho, \sigma) = E_{\mathbf{y}} \left[ \sum_{i=1}^N f \left( \frac{\delta(\mathbf{x}_i, \mathbf{x})}{\rho} \right) \left( 1 - \exp \left( -\frac{\|\mathbf{y}_i - \mathbf{y}\|_2^2}{2\sigma^2} \right) \right) \sigma^2 \right], \quad (1.38)$$

where  $E_{\mathbf{y}}$  is the expectation with respect to  $\mathbf{y}$ ,  $\delta(\mathbf{x}_i, \mathbf{x})$  is a distance between  $\mathbf{x}_i$  and  $\mathbf{x}$ , and  $f$  is a monotonically decreasing function. Parameters  $\rho$  and  $\sigma$  are bandwidths that determine the neighborhood radii in each space. The scaled upside-down Gaussian bell  $\sigma^2(1 - \exp(-u^2/\sigma^2))$  introduces adjustable saturation into the quadratic factor  $u^2$ . The gradient of the cost function can be written as

$$\frac{\partial E(\mathbf{Y}; \mathbf{X}, \rho, \sigma)}{\partial \mathbf{y}_i} = E_{\mathbf{y}} \left[ f \left( \frac{\delta(\mathbf{x}_i, \mathbf{x})}{\rho} \right) \exp \left( -\frac{\|\mathbf{y}_i - \mathbf{y}\|_2^2}{2\sigma^2} \right) (\mathbf{y}_i - \mathbf{y}) \right]. \quad (1.39)$$

If we adopt a stochastic gradient descent, we can drop the expectation and write the update

$$\mathbf{y}_i \leftarrow \mathbf{y}_i - \alpha f \left( \frac{\delta(\mathbf{x}_i, \mathbf{x})}{\rho} \right) \exp \left( -\frac{\|\mathbf{y}_i - \mathbf{y}\|_2^2}{2\sigma^2} \right) (\mathbf{y}_i - \mathbf{y}), \quad (1.40)$$

where  $\alpha$  is a slowly decreasing step size. In practice, we still need to know the distribution of  $\mathbf{y}$  and the value of  $\mathbf{x}$  for a given  $\mathbf{y}$ . We can approximate the distribution with a mixture of unit variance Gaussians, namely

$$p(\mathbf{y}) = \frac{1}{N} \sum_{i=1}^N \frac{1}{\sqrt{2\pi}} \exp(-\|\mathbf{y} - \mathbf{y}_i\|_2^2/2), \quad (1.41)$$

and draw randomly generated instances of  $\mathbf{y}$  from it. The distance between  $\mathbf{x}_i$  and  $\mathbf{x}$  can be approximated by  $\delta(\mathbf{x}_i, \mathbf{x}) \approx \delta(\mathbf{x}_i, \mathbf{x}_j) = \delta_{ij}(\mathbf{X})$ , where  $j = \operatorname{argmin}_i \|\mathbf{y} - \mathbf{y}_i\|_2^2$ . All distances  $\delta_{ij}(\mathbf{X})$  in the high-dimensional space are known and can be Euclidean or correspond to the shortest paths in a neighborhood graph, like in methods described in previous sections. In the case of shortest paths, the difference with a SOM becomes obvious: the graph is established in the high-dimensional data space in a data-driven way, whereas the user arbitrarily defines it in the low-dimensional visualization space. Apart from this key difference, Isotop and a SOM share many common features. In both methods, coordinates are updated in an iterative way and the amplitude of the update is modulated by a factor that depends on distances in a graph.

As most graph layout techniques are based on heuristic approaches, there are of course many possible variants, such as the exploratory observation machine (XOM) [43].

## 1.8 Examples and comparisons

### 1.8.1 Quality assessment

Dimensionality reduction aims at providing a faithful representation of data. The algorithms described in this chapter, among others, provide ways to obtain such a representation. However it is not obvious for the user to decide which method best suits the problem. An objective evaluation of the data representation quality is necessary. Nevertheless, using the cost function of a specific method as quality criterion obviously biases any comparison result towards the chosen specific method. There is thus a need for criteria that are as independent as possible from all methods.

Intuitively, a good representation should preserve neighborhood relationships: close points in the data space should remain near each other in the representation, while distant points should remain far from each other. This idea could be implemented very simply with a distance preservation criterion, such as Sammon's stress [8]. However, distance criteria prove to be too strict, since neighborhood relationships can be preserved even if distances are stretched or shrunk. For this reason, modern quality criteria [68, 69, 70, 71, 72] involve sorted distances and ranked neighbors.

Let us define the rank of  $\mathbf{x}_j$  with respect to  $\mathbf{x}_i$  in the high-dimensional space as  $r_{ij}(\mathbf{X}) = |\{k : \delta_{ik}(\mathbf{X}) < \delta_{ij}(\mathbf{X}) \text{ or } (\delta_{ik}(\mathbf{X}) = \delta_{ij}(\mathbf{X}) \text{ and } 1 \leq k < j \leq N)\}|$ , where  $|\mathcal{A}|$  denotes the cardinality of set  $\mathcal{A}$ . Intuitively, the rank counts the number of points that are closer neighbors of  $\mathbf{x}_i$  than  $\mathbf{x}_j$ , including  $\mathbf{x}_i$  itself. The ranking is established according to some given distances (Euclidean in the following experiments) and ties are circumvented by switching to point indices. Similarly, the rank of  $\mathbf{y}_j$  with respect to  $\mathbf{y}_i$  in the low-dimensional space is  $r_{ij}(\mathbf{Y}) = |\{k : \delta_{ik}(\mathbf{Y}) < \delta_{ij}(\mathbf{Y}) \text{ or } (\delta_{ik}(\mathbf{Y}) = \delta_{ij}(\mathbf{Y}) \text{ and } 1 \leq k < j \leq N)\}|$ . Hence, reflexive ranks are set to zero ( $r_{ii}(\mathbf{X}) = r_{ii}(\mathbf{Y}) = 0$ ) and ranks are unique, i.e. there are no *ex aequo* ranks:  $r_{ij}(\mathbf{X}) \neq r_{ik}(\mathbf{X})$  for  $k \neq j$ , even if  $\delta_{ij}(\mathbf{X}) = \delta_{ik}(\mathbf{X})$ . This means that nonreflexive ranks belong to  $\{1, \dots, N-1\}$ . The nonre-

flexive  $K$ -ary neighborhoods of  $\mathbf{x}_i$  and  $\mathbf{y}_i$  are denoted by  $\mathcal{N}_i^K(\mathbf{X}) = \{j : 1 \leq r_{ij}(\mathbf{X}) \leq K\}$  and  $\mathcal{N}_i^K(\mathbf{Y}) = \{j : 1 \leq r_{ij}(\mathbf{Y}) \leq K\}$ , respectively.

The co-ranking matrix [70] can then be defined as

$$\mathbf{Q} = [q_{kl}]_{1 \leq k, l \leq N-1} \quad \text{with} \quad q_{kl} = |\{(i, j) : r_{ij}(\mathbf{X}) = k \text{ and } r_{ij}(\mathbf{Y}) = l\}| . \quad (1.42)$$

The co-ranking matrix is the joint histogram of the ranks and is actually a sum of  $N$  permutation matrices of size  $N - 1$ . With an appropriate gray scale, the co-ranking matrix can also be displayed and interpreted in a similar way as a Shepard diagram [5]. Historically, this scatterplot has often been used to assess results of multidimensional scaling and related methods [10]; it shows the distances  $\delta_{ij}(\mathbf{X})$  with respect to the corresponding distances  $\delta_{ij}(\mathbf{Y})$ , for all pairs  $(i, j)$ , with  $i \neq j$ . The analogy between the co-ranking matrix and Shepard's diagram suggests that meaningful criteria should focus on the upper and lower triangle of the co-ranking matrix  $\mathbf{Q}$ . This can be done by considering the pair of criteria proposed in [71, 72]. They are defined as

$$Q_{\text{NX}}(K) = \frac{1}{KN} \sum_{k=1}^K \sum_{l=1}^K q_{kl} \quad (1.43)$$

and

$$B_{\text{NX}}(K) = \frac{1}{KN} \sum_{k=1}^K \left( \sum_{l=k+1}^K q_{kl} - \sum_{l=1}^{k-1} q_{kl} \right) . \quad (1.44)$$

The first criterion assesses the overall quality of the embedding and varies between 0 and 1. It measures the average agreement between the corresponding neighborhoods in the high- and low-dimensional spaces. To see this,  $Q_{\text{NX}}(K)$  can be rewritten as

$$Q_{\text{NX}}(K) = \frac{1}{KN} \sum_{i=1}^N |\mathcal{N}_i^K(\mathbf{X}) \cap \mathcal{N}_i^K(\mathbf{Y})| . \quad (1.45)$$

The second criterion assesses the balance between the upper and lower triangles of the upper left  $K$ -by- $K$  block of the co-ranking matrix. This criterion is useful to distinguish two different types of errors: distant points in the data space that become erroneously neighbors in the visualization space and neighbors that are mistakenly represented too far away. In the case of a nonlinear manifold, the first error type occurs e.g. if two distant patches are superimposed in the visualization ( $B_{\text{NX}}(K)$  will be positive), whereas the second error type happens e.g. with torn patches ( $B_{\text{NX}}(K)$  will be negative).

## 1.8.2 Data sets

Two typical data sets illustrate the methods described in the previous sections. The first one is an academic example based on the so-called Swiss roll [16], which is a 2D manifold embedded in a 3D space. The Swiss roll basically looks like a rectangular piece of plane curved as a spiral. The parametric equations are:

$$\mathbf{x} = [\sqrt{u} \cos(3\pi\sqrt{u}), \sqrt{u} \sin(3\pi\sqrt{u}), \pi v]^T , \quad (1.46)$$

where  $u$  and  $v$  are uniformly distributed between 0 and 1. The distribution of  $\mathbf{x}$  on the Swiss roll is uniform as well. To add some difficulty to the exercise, a disc in the center of the curved rectangle is removed, as shown in Fig. 1.1 (left). There are approximately 950 points in the data set.

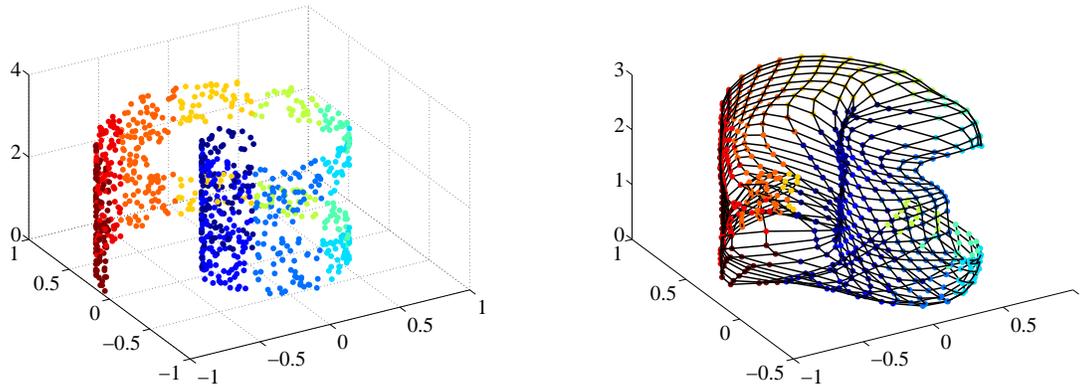


Figure 1.1: (left) About 950 points sampled from a Swiss roll with a hole. The color varies with respect to the spiral radius. (right) A 20-by-30 SOM learned on the Swiss roll sample.

The second data set includes about 1000 images of handwritten digits. They are randomly drawn from the MNIST data base [73]. Typical images are shown in Fig. 1.2. Each image contains  $28^2$  pixels with a gray

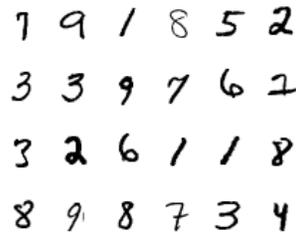


Figure 1.2: Examples of scanned handwritten digits randomly drawn from the MNIST database. Each image is 28 pixels wide and 28 pixels high. Gray levels range from 0 to 1.

level ranging from 0 to 1. All images are converted into 784-dimensional vectors. Classical metric MDS is used to preprocess the data set: 97.5% of the total variance is preserved, leading to a reduced dimensionality about 200, depending on the random sample. This allows a first drastic reduction of the dimensionality with almost no loss of information or structure, which is intended to accelerate all subsequent computations.

For both data sets, two-dimensional representations are computed with NLDR methods. In the case of the Swiss roll, the embedding dimensionality is thus in agreement with the intrinsic dimensionality of the underlying manifold. In contrast, the degrees of freedom in the MNIST data base are expected to be much

more numerous and a good 2D visualization is much more difficult to obtain.

### 1.8.3 Methods

The compared methods are CMMDS, Sammon's NLM, CCA, Isotop, a 20-by-30 SOM, LLE, Laplacian eigenmaps, and MVU. The first four methods are used successively with Euclidean distances and geodesic distances. The latter are computed with  $K$ -ary neighborhoods, where  $K = 7$ . All data points outside of the largest connected component are discarded. The same value of  $K$  is used for other methods involving  $K$ -ary neighborhoods (LLE and Laplacian eigenmaps). In order to reduce the huge computational cost of MVU, we used the faster variant described in [74], with  $K = 5$  and 5% of data points randomly chosen to be landmarks. All methods run with default parameter values if not otherwise specified.

### 1.8.4 Results

Figure 1.3 shows a visual representation of the two-dimensional embeddings of the Swiss roll obtained with the twelve considered methods. In the particular case of the SOM, only grid nodes have coordinates both in the high- and low-dimensional space. As a simple workaround to this limitation, data points are given the same coordinates as their closest grid nodes in the high-dimensional space. Grid nodes that are never selected as the closest one are shown as a blank cell. The configuration of the SOM in the data space is visible in Fig. 1.1 (right).

The quality assessment curves are shown in Fig. 1.4, for a neighborhood size  $K$  ranging from 1 to 300. The solid and dashed lines refer to  $Q_{NX}(K)$  (above, with baseline  $K/(N-1)$ ) and  $B_{NX}(K)$  (below, with baseline 0), respectively. As can be seen, for distance-based methods, the use of graph distances yield a better unfolding of the Swiss roll than Euclidean ones. Among these methods, CCA performs the best, followed by Sammon's NLM and finally CMMDS. This shows that a global minimum found by a spectral optimization technique such as in CMMDS does not systematically outperform methods relying on gradient descent. Other spectral methods, such as LLE, Laplacian eigenmaps, and MVU work well though the quality of their results is slightly lower. In particular, MVU suffers from convergence issues in its semidefinite programming step (the constraints are apparently too tight and no solution is found). The SOM achieves an intermediate performance; due to the inherent vector quantization in this method, small neighborhoods ( $K < 20$ ) are not well preserved (rank information is lost for all data points sharing the same closest grid node). Back to Fig. 1.3, we see that methods showing a negative value for  $B_{NX}(K)$  are those that are able to actually unfold the Swiss roll. Among those methods, CCA with graph distance is the only method able to faithfully render the true latent space of the Swiss roll: a rectangle with a circular hole. Other distance-based methods tend to stretch the hole; this effect is caused by an overestimation of the distances,

as geodesic paths are forced to circumvent the hole.

As to the handwritten digit images, the 2D visualizations computed with the various NLDR methods are shown in Fig. 1.5. Each visualization consists of an array of 20-by-20 bins decorated with thumbnail images. Each thumbnail is the average of all images falling in the corresponding bin. If dissimilar images are gathered in the same bin, the thumbnail looks blurred. Empty bins are left blank. The shape of the embedding provided by the SOM trivially depends on the chosen grid. As can be seen, methods relying on (stochastic) gradient descent, such as Sammon's NLM, CCA, and Isotop, yield disc-shaped embeddings. In contrast, spectral methods involving a sparse similarity matrix such as LLE and Laplacian eigenmaps tend to produce spiky embeddings. Knowing that the data set is likely to be clustered (one cluster for each of the ten digits), the objective functions of these methods tend to increase the distances between the clusters. In Laplacian eigenmaps this effect results from minimizing the distances between neighbors, while keeping a fixed variance for all points; this remains valid for LLE, due to their close relationship. Maximal distances from one cluster to all others are obtained in a hyper-pyramidal configuration; for ten clusters, such a configuration spans at least nine dimensions. The linear projection of a hyper-pyramid look exactly like the spiky embeddings observed in Fig. 1.5, with three corners correctly represented in two dimensions and all others collapsed in the center. This reasoning is indirectly confirmed by looking at the eigenvalue spectrum involved in these methods, showing that variance actually spreads out across many dimensions. This is a fundamental difference between spectral and non-spectral methods: the former prune dimensions only after nonlinear transformation of data, whereas the latter are directly forced to work in a low-dimensional space.

Quality assessment curves are displayed in Fig. 1.6. As expected, the overall performance level is lower than for the Swiss roll, as the intrinsic dimensionality of data is much higher than two. For small values of  $K$ , hardly half of the neighbors are preserved, as indicated by  $Q_{NX}(K)$ . The best two methods are CCA with graph distances and the SOM; they outperform all others by far. For small neighborhood sizes, the SOM is hindered by its inherent vector quantization. As with the Swiss roll, the methods with the highest values of  $Q_{NX}(K)$  are also those with the lower  $B_{NX}(K)$ . The third method is Isotop with graph distances. In this example, Isotop is much faster than the SOM, because the computation of closest points is much cheaper in the low-dimensional visualization space than in the high-dimensional data space. Spectral methods yield lower performances. The semidefinite programming step in MVU does not succeed in finding a satisfying solution. Sammon's NLM with Euclidean distances performs the worst and also hardly converges.

## 1.9 Conclusions

Dimensionality reduction proves to be a powerful tool for data visualization and exploratory analysis. From early linear techniques such as PCA and CMMDS to modern nonlinear methods, more than a century has

passed. As shown in this chapter, the use of graphs is an important breakthrough in the domain and largely contributes to a significant performance leap. Methods relying on graphs can be categorized with respect to mainly two criteria, namely the data properties they consider (dissimilarities or similarities) and their optimization technique (spectral or non-spectral). For instance, Isomap and MVU involve graph distances (shortest paths and specifically optimized distances, respectively) and a spectral optimization (CMMDS). Sammon's nonlinear mapping and CCA can use shortest paths as well, but they work with a pseudo-Newton optimization and stochastic gradient descent, respectively. Spectral methods based on similarities are for instance Laplacian eigenmaps, LLE, and their variants. Isotop and the SOM also entail similarities, but they rely on heuristic optimization schemes. Spectral methods provide strong theoretical guarantees, such as the capability to find a global minimum. In practice, however, non-spectral techniques often outperform them, thanks to a greater flexibility and the possibility to handle more relevant or more complex cost functions.

If distance preservation has been extensively investigated for quite some time, the use of similarities remains largely unexplored. Spectral methods that uses sparse matrices of pairwise similarities, such as Laplacian eigenmaps and LLE, eventually seem to be more suited to clustering problems than to dimensionality reduction. On the other hand, non-spectral methods like Isotop and the SOM lack sound theoretical foundations or are impeded by their inherent vector quantization. In the near future, progress is likely to stem from more elaborate similarity preservation schemes, such as those developed in [75, 76, 77]. There is a hope that while naturally enforcing the preservation of the local structure, carefully designed similarities [78] can also address important issues such as the phenomenon of distance concentration [79].

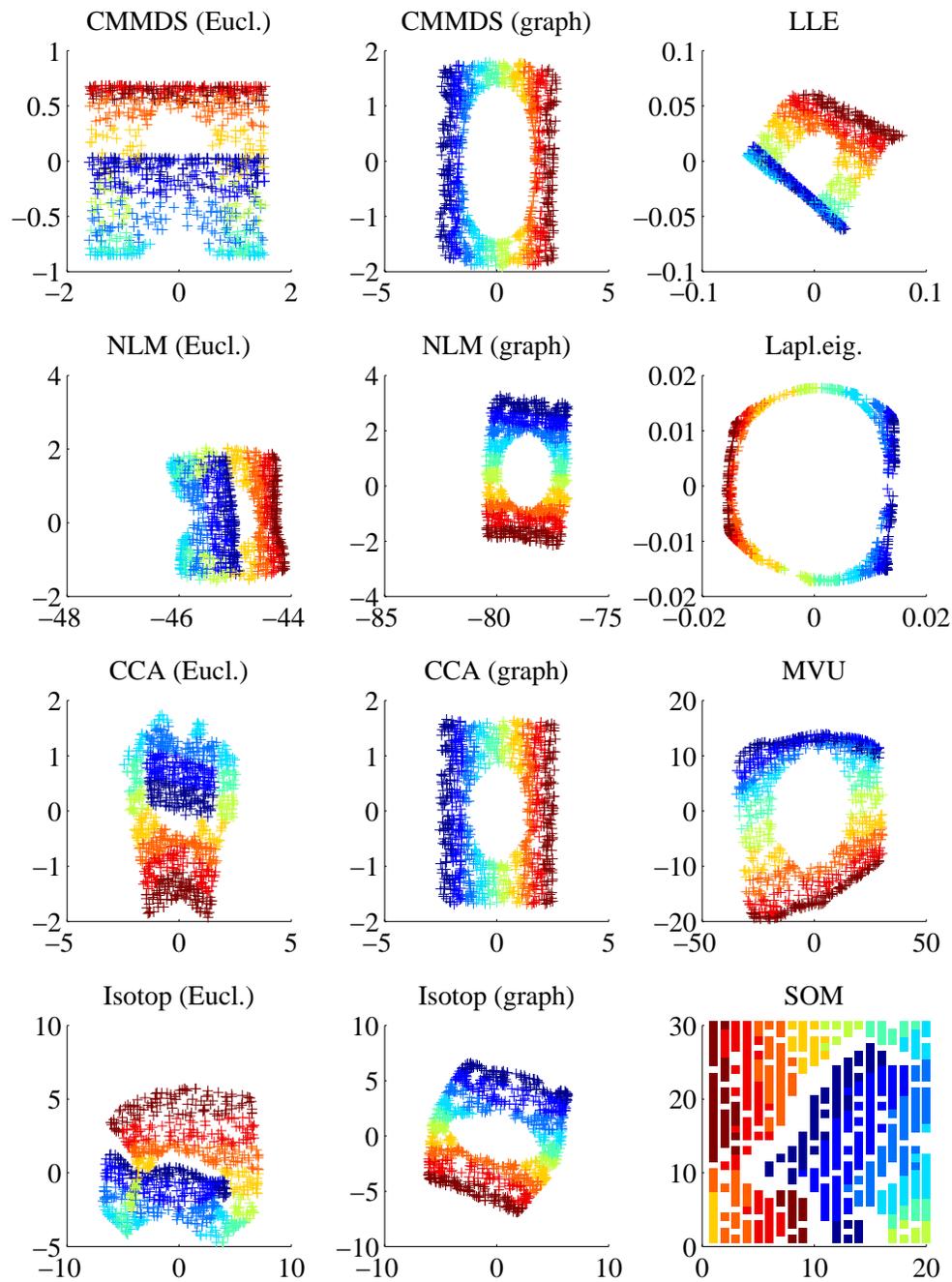


Figure 1.3: Two-dimensional embeddings of the Swiss roll with various NLD methods. Distance-based methods use either Euclidean norm ('Eucl.') or shortest paths in a neighborhood graph ('Graph').

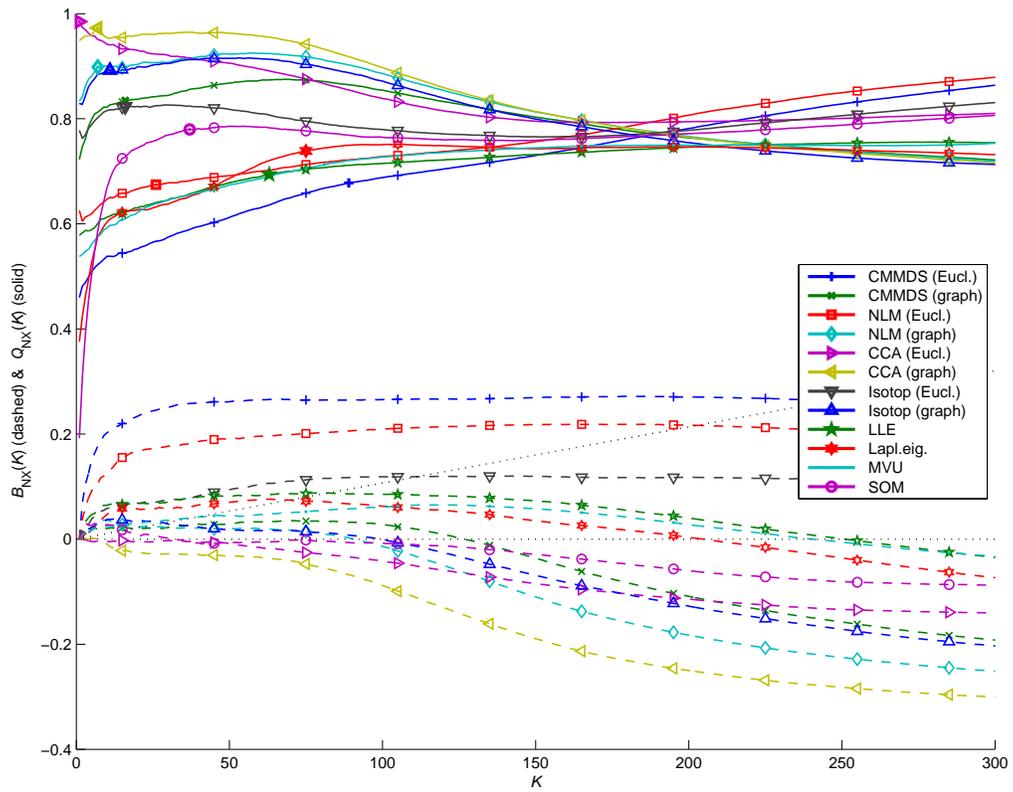


Figure 1.4: Quality assessment curves for the two-dimensional embeddings of the Swiss roll with various NLDR methods.

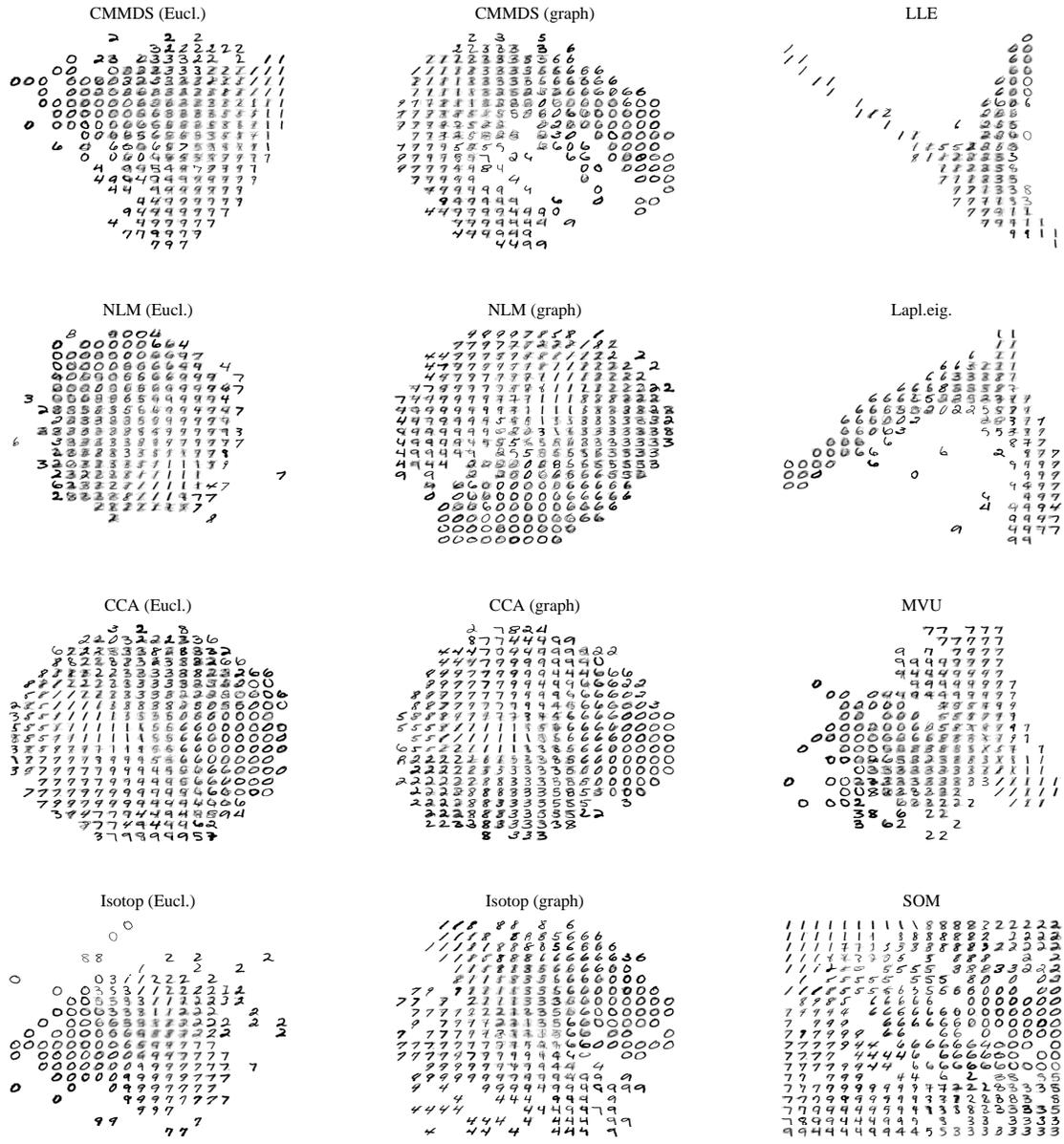


Figure 1.5: Two-dimensional embeddings of the handwritten digits with various NLDR methods. Distance-based methods use either Euclidean norm ('Eucl.') or shortest paths in a neighborhood graph ('Graph').

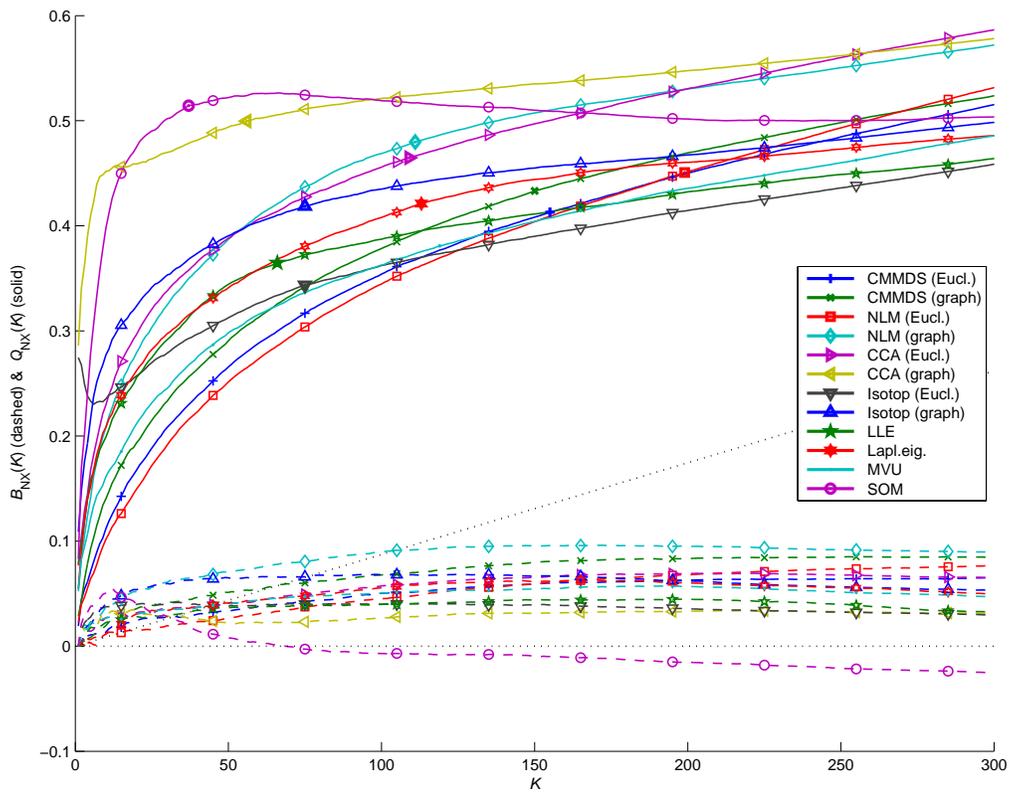


Figure 1.6: Quality assessment curves for the two-dimensional embeddings of the handwritten digits with various NLDR methods.



# Bibliography

- [1] I. Jolliffe, *Principal Component Analysis*. New York, NY: Springer-Verlag, 1986.
- [2] G. Young and A. Householder, “Discussion of a set of points in terms of their mutual distances,” *Psychometrika*, vol. 3, pp. 19–22, 1938.
- [3] W. Torgerson, “Multidimensional scaling, I: Theory and method,” *Psychometrika*, vol. 17, pp. 401–419, 1952.
- [4] J. Lee and M. Verleysen, *Nonlinear dimensionality reduction*. Springer, 2007.
- [5] R. Shepard, “The analysis of proximities: Multidimensional scaling with an unknown distance function (parts 1 and 2),” *Psychometrika*, vol. 27, pp. 125–140, 219–249, 1962.
- [6] J. Kruskal, “Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis,” *Psychometrika*, vol. 29, pp. 1–28, 1964.
- [7] Y. Takane, F. Young, and J. de Leeuw, “Nonmetric individual differences multidimensional scaling: an alternating least squares method with optimal scaling features,” *Psychometrika*, vol. 42, pp. 7–67, 1977.
- [8] J. Sammon, “A nonlinear mapping algorithm for data structure analysis,” *IEEE Transactions on Computers*, vol. CC-18, no. 5, pp. 401–409, 1969.
- [9] P. Demartines and J. Héroult, “Vector quantization and projection neural network,” ser. Lecture Notes in Computer Science, A. Prieto, J. Mira, and J. Cabestany, Eds. New York: Springer-Verlag, 1993, vol. 686, pp. 328–333.
- [10] ———, “Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets,” *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 148–154, Jan. 1997.
- [11] T. Kohonen, “Self-organization of topologically correct feature maps,” *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.
- [12] M. Kramer, “Nonlinear principal component analysis using autoassociative neural networks,” *AIChE Journal*, vol. 37, no. 2, pp. 233–243, 1991.
- [13] E. Oja, “Data compression, feature extraction, and autoassociation in feedforward neural networks,” in *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, Eds. North-Holland: Elsevier Science Publishers, B.V., 1991, vol. 1, pp. 737–745.

- [14] J. Mao and A. Jain, “Artificial neural networks for feature extraction and multivariate data projection,” *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 296–317, Mar. 1995.
- [15] B. Schölkopf, A. Smola, and K.-R. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” *Neural Computation*, vol. 10, pp. 1299–1319, 1998.
- [16] J. Tenenbaum, V. de Silva, and J. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
- [17] S. Roweis and L. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
- [18] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, June 2003.
- [19] D. Donoho and C. Grimes, “Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data,” in *Proceedings of the National Academy of Arts and Sciences*, vol. 100, 2003, pp. 5591–5596.
- [20] K. Weinberger and L. Saul, “Unsupervised learning of image manifolds by semidefinite programming,” *International Journal of Computer Vision*, vol. 70, no. 1, pp. 77–90, 2006.
- [21] L. Xiao, J. Sun, and S. Boyd, “A duality view of spectral methods for dimensionality reduction,” in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburg, PA, 2006, pp. 1041–1048.
- [22] L. Saul and S. Roweis, “Think globally, fit locally: Unsupervised learning of nonlinear manifolds,” *Journal of Machine Learning Research*, vol. 4, pp. 119–155, June 2003.
- [23] N. Linial, E. London, and Y. Rabinovich, “The geometry of graphs and some of its algorithmic applications,” *Combinatorica*, vol. 15, no. 2, pp. 215–245, 1995.
- [24] G. Di Battista, P. Eades, R. Tamassia, and I. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1998.
- [25] I. Herman, G. Melançon, and M. Marshall, “Graph visualization and navigation in information visualization: A survey,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 6, pp. 24–43, 2000.
- [26] Y. Bengio, P. Vincent, J.-F. Paiement, O. Delalleau, M. Ouimet, and N. Le Roux, “Spectral clustering and kernel PCA are learning eigenfunctions,” Département d’Informatique et Recherche Opérationnelle, Université de Montréal, Montréal, Tech. Rep. 1239, July 2003.
- [27] M. Saerens, F. Fouss, L. Yen, and P. Dupont, “The principal components analysis of a graph, and its relationships to spectral clustering,” in *Proceedings of the 15th European Conference on Machine Learning (ECML 2004)*, 2004, pp. 371–383.
- [28] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, “Diffusion maps, spectral clustering and eigenfunction of Fokker-Planck operators,” in *Advances in Neural Information Processing Systems (NIPS 2005)*, Y. Weiss, B. Schölkopf, and J. Platt, Eds. Cambridge, MA: MIT Press, 2006, vol. 18.

- [29] M. Brand and K. Huang, “A unifying theorem for spectral embedding and clustering,” in *Proceedings of International Workshop on Artificial Intelligence and Statistics (AISTATS’03)*, C. Bishop and B. Frey, Eds., Jan. 2003.
- [30] K. Pearson, “On lines and planes of closest fit to systems of points in space,” *Philosophical Magazine*, vol. 2, pp. 559–572, 1901.
- [31] H. Hotelling, “Analysis of a complex of statistical variables into principal components,” *Journal of Educational Psychology*, vol. 24, pp. 417–441, 1933.
- [32] K. Karhunen, “Zur Spektraltheorie stochastischer Prozesse,” *Ann. Acad. Sci. Fennicae*, vol. 34, 1946.
- [33] M. Loève, “Fonctions aléatoire du second ordre,” in *Processus stochastiques et mouvement Brownien*, P. Lévy, Ed. Paris: Gauthier-Villars, 1948, p. 299.
- [34] A. Kearsley, R. Tapia, and M. Trosset, “The solution of the metric STRESS and SSTRESS problems in multidimensional scaling using newton’s method,” *Computational Statistics*, vol. 13, no. 3, pp. 369–396, 1998.
- [35] J. Lee, A. Lendasse, N. Donckers, and M. Verleysen, “A robust nonlinear projection method,” in *Proceedings of ESANN 2000, 8th European Symposium on Artificial Neural Networks*, M. Verleysen, Ed. Bruges, Belgium: D-Facto public., Apr. 2000, pp. 13–20.
- [36] J. Lee and M. Verleysen, “Curvilinear distance analysis versus isomap,” *Neurocomputing*, vol. 57, pp. 49–76, Mar. 2004.
- [37] J. Peltonen, A. Klami, and S. Kaski, “Learning metrics for information visualisation,” in *Proceedings of the 4th Workshop on Self-Organizing Maps (WSOM’03)*, Hibikino, Kitakyushu, Japan, Sept. 2003, pp. 213–218.
- [38] L. Yang, “Sammon’s nonlinear mapping using geodesic distances,” in *Proc. 17th International Conference on Pattern Recognition (ICPR’04)*, 2004, vol. 2.
- [39] P. Estévez and A. Chong, “Geodesic nonlinear mapping using the neural gas network,” in *Proceedings of IJCNN 2006*, 2006, in press.
- [40] M. Belkin and P. Niyogi, “Laplacian eigenmaps and spectral techniques for embedding and clustering,” in *Advances in Neural Information Processing Systems (NIPS 2001)*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2002, vol. 14.
- [41] T. Kohonen, *Self-Organizing Maps*, 2nd ed. Heidelberg: Springer, 1995.
- [42] J. Lee, C. Archambeau, and M. Verleysen, “Locally linear embedding versus Isotop,” in *Proceedings of ESANN 2003, 11th European Symposium on Artificial Neural Networks*, M. Verleysen, Ed. Bruges, Belgium: d-side, Apr. 2003, pp. 527–534.
- [43] A. Wismüller, “The exploration machine - a novel method for data visualization,” in *Lecture Notes in Computer Science. Advances in Self-Organizing Maps*, 2009, pp. 344–352.
- [44] M. Bernstein, V. de Silva, J. Langford, and J. Tenenbaum, “Graph approximations to geodesics on embedded manifolds,” Stanford University, Palo Alto, CA, Tech. Rep., Dec. 2000.

- [45] E. Dijkstra, "A note on two problems in connection with graphs," *Numerical Mathematics*, vol. 1, pp. 269–271, 1959.
- [46] M. Fredman and R. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *Journal of the ACM*, vol. 34, pp. 596–615, 1987.
- [47] F. Shang, L. Jiao, J. Shi, and J. Chai, "Robust positive semidefinite l-isomap ensemble," *Pattern Recognition Letters*, vol. 32, no. 4, pp. 640–649, 2011.
- [48] H. Choi and S. Choi, "Robust kernel isomap," *Pattern Recognition*, vol. 40, no. 3, pp. 853–862, 2007.
- [49] K. Weinberger and L. Saul, "Unsupervised learning of image manifolds by semidefinite programming," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-04)*, vol. 2, Washington, DC, 2004, pp. 988–995.
- [50] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
- [51] L. Yen, D. Vanvyve, F. Wouters, F. Fouss, M. Verleysen, and M. Saerens, "Clustering using a random-walk based distance measure," in *Proceedings of ESANN 2005, 13th European Symposium on Artificial Neural Networks*, M. Verleysen, Ed. Bruges, Belgium: d-side, Apr. 2005, pp. 317–324.
- [52] L. Grady and J. Polimeni, *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. New York: Springer, 2010.
- [53] G. Daza-Santacoloma, C. Acosta-Medina, and G. Castellanos-Dominguez, "Regularization parameter choice in locally linear embedding," *Neurocomputing*, vol. 73, no. 10–12, pp. 1595–1605, 2010.
- [54] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, pp. 84–95, 1980.
- [55] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I: Statistics*, L. Le Cam and J. Neyman, Eds. Berkeley and Los Angeles, CA: University of California Press, 1967, pp. 281–297.
- [56] E. Erwin, K. Obermayer, and K. Schulten, "Self-organizing maps: ordering, convergence properties and energy functions," *Biological Cybernetics*, vol. 67, pp. 47–55, 1992.
- [57] A. Ultsch, "Maps for the visualization of high-dimensional data spaces," in *Proc. Workshop on Self-Organizing Maps (WSOM 2003)*, Kyushu, Japan, 2003, pp. 225–230.
- [58] H.-U. Bauer, M. Herrmann, and T. Villmann, "Neural maps and topographic vector quantization," *Neural Networks*, vol. 12, pp. 659–676, 1999.
- [59] G. Goodhill and T. Sejnowski, "Quantifying neighbourhood preservation in topographic mappings," in *Proceedings of the Third Joint Symposium on Neural Computation*. University of California, Pasadena, CA: California Institute of Technology, 1996, pp. 61–82.

- [60] H.-U. Bauer and K. Pawelzik, “Quantifying the neighborhood preservation of self-organizing maps,” *IEEE Transactions on Neural Networks*, vol. 3, pp. 570–579, 1992.
- [61] M. de Bodt, E. Cottrell and M. Verleysen, “Statistical tools to assess the reliability of self-organizing maps,” *Neural Networks*, vol. 15, no. 8–9, pp. 967–978, 2002.
- [62] K. Kiviluoto, “Topology preservation in self-organizing maps,” in *Proc. Int. Conf. on Neural Networks, ICNN’96*, I. N. N. Council, Ed., vol. 1, Piscataway, NJ, 1996, pp. 294–299, also available as technical report A29 of the Helsinki University of Technology.
- [63] C. Bishop, M. Svensén, and K. Williams, “GTM: A principled alternative to the self-organizing map,” *Neural Computation*, vol. 10, no. 1, pp. 215–234, 1998.
- [64] T. Martinetz and K. Schulten, “A “neural-gas” network learns topologies,” in *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, Eds. Amsterdam: Elsevier, 1991, vol. 1, pp. 397–402.
- [65] —, “Topology representing networks,” *Neural Networks*, vol. 7, no. 3, pp. 507–522, 1994.
- [66] M. Jünger and P. Mutzel, *Graph Drawing Software*. Springer-Verlag, 2004.
- [67] J. Lee and M. Verleysen, “Nonlinear projection with the Isotop method,” in *LNCS 2415: Artificial Neural Networks, Proceedings of ICANN 2002*, J. Dorronsoro, Ed. Madrid (Spain): Springer, Aug. 2002, pp. 933–938.
- [68] J. Venna and S. Kaski, “Neighborhood preservation in nonlinear projection methods: An experimental study,” in *Proceedings of ICANN 2001*, G. Dorffner, H. Bischof, and K. Hornik, Eds. Berlin: Springer, 2001, pp. 485–491.
- [69] L. Chen and A. Buja, “Local multidimensional scaling for nonlinear dimension reduction, graph drawing, and proximity analysis,” *Journal of the American Statistical Association*, vol. 101, no. 485, pp. 209–219, 2009.
- [70] J. Lee and M. Verleysen, “Rank-based quality assessment of nonlinear dimensionality reduction,” in *Proceedings of ESANN 2008, 16th European Symposium on Artificial Neural Networks*, M. Verleysen, Ed. Bruges: d-side, Apr. 2008, pp. 49–54.
- [71] —, “Quality assessment of nonlinear dimensionality reduction based on k-ary neighborhoods,” in *JMLR Workshop and Conference Proceedings (New challenges for feature selection in data mining and knowledge discovery)*, Y. Saeys, H. Liu, I. Inza, L. Wehenkel, and Y. Van de Peer, Eds., Sept. 2008, vol. 4, pp. 21–35.
- [72] —, “Quality assessment of dimensionality reduction: Rank-based criteria,” *Neurocomputing*, vol. 72, no. 7–9, pp. 1431–1443, 2009.
- [73] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [74] K. Weinberger, B. Packer, and L. Saul, “Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization,” in *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*, R. Cowell and Z. Ghahramani, Eds. Bardados: Society for Artificial Intelligence and Statistics, Jan. 2005, pp. 381–388.

- [75] G. Hinton and S. Roweis, “Stochastic neighbor embedding,” in *Advances in Neural Information Processing Systems (NIPS 2002)*, S. Becker, S. Thrun, and K. Obermayer, Eds. MIT Press, 2003, vol. 15, pp. 833–840.
- [76] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.
- [77] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski, “Information retrieval perspective to nonlinear dimensionality reduction for data visualization,” *Journal of Machine Learning Research*, vol. 11, pp. 451–490, 2010.
- [78] J. A. Lee and M. Verleysen, “Shift-invariant similarities circumvent distance concentration in stochastic neighbor embedding and variants,” in *Proc. International Conference on Computational Science (ICCS 2011)*, Singapore, 2011.
- [79] D. François, V. Wertz, and M. Verleysen, “The concentration of fractional distances,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 7, pp. 873–886, July 2007.

# Index

- affinity, 8
- ball, 7, 9
- bandwidth, 16, 17
- Bayesian, 16
- CCA, *see* component, 6, 9, 21, 23
- centered, 5
- centering
  - double, 5
  - matrix, 5, 13
- centroid, 17
- cluster, 2, 3, 23
  - spectral clustering, 3
- CMMDS, *see* multidimensional scaling, 6, 7, 9–11, 13, 15, 20–22
- co-ranking matrix, 19
- component
  - connected, 12, 15, 21
  - curvilinear, 3, 6, 9
  - principal, 2, 4
- concentration, 23
- conductance, 13
- constraint, 10–12, 14
- coordinate, 5, 6, 10, 15–18
  - centered, 5
  - nonlinearly transformed, 13
- coordinates, 14
- cost function, 6, 7, 12, 14, 17, 18, 23
- covariance, 4, 5
- curvature, 8
- data
  - high-dimensional, 1, 2
  - representation, 2, 18
  - set, 19, 20
  - visualization, 2, 15, 16, 22
- decoding, 3
- decorrelation, 5
- derivative
  - partial, 4, 12
- diffusion, 13
- Dijkstra, 9–11
- dimensionality
  - curse of, 2
  - intrinsic, 20
  - reduction, 1–3, 15, 18, 22, 23
    - linear, 2, 3
    - nonlinear, 2
- dissimilarity, 2, 11, 23
- distance, 1, 3, 7, 15, 17, 19, 23
  - commute time, 2, 13
  - commute-time, 13
  - concentration, 23
  - diffusion, 13
  - Euclidean, 5, 8–11, 21
    - squared, 6, 9
  - geodesic, 2, 8–10, 13, 21
  - graph, 7, 9, 10, 22
  - infinite, 10, 15
  - matrix
    - dense, 13
    - squared, 5, 9
  - preservation, 6, 8, 9, 18, 23
    - weighted, 9
  - sorted, 18
- distortion, 16
- distribution, 15, 17, 20
  - support, 17
- dot product, 2, 6, 13
- DR, *see* dimensionality, 2, 3, 10, 11, 17

- linear, 2
- eigenproblem, 4
- eigenspectrum, 10
- eigenvalue, 4, 5, 9, 11, 22
  - multiplicity, 12, 15
  - problem, 12
- eigenvector, 5, 15
  - leading, 5
  - trailing, 12, 15
- embedding, 1, 3, 9, 11, 14, 17
  - graph, 3, 8
  - locally linear, 8, 14
  - spiky, 22
- encoding, 3
- expectation, 17
- expectation-maximization, 16
- exploratory
  - analysis, 22
  - data analysis, 16
  - observation machine, 8, 18
- Floyd, 9–11
- force, 8
- geodesic
  - curve, 8
  - distance, *see* distance
- gradient, 17
  - descent, 6
    - stochastic, 7, 17, 22, 23
- Gram matrix, 2, 5, 9, 10, 12, 13, 15
- graph, 3, 7, 11, 23
  - construction, 8, 9, 15
  - distance, 7
  - drawing, 17
  - edge, 13
  - embedding, 8, 15, 17
  - layout, 17, 18
  - placement, 8
  - undirected, 12, 15
  - vertex, 17
- grid, 15
- elasticity, 16
- node, 16, 17
- handwritten digit, 2, 20
- histogram
  - joint, 19
- honeycomb, 16
- Isomap, 7, 9–11, 13
- isometry, 9
- Isotop, 8, 17, 18, 21, 23
- K-means, 16, 17
- kernel, 13
- Lagrange, 4
  - multiplier, 4
- Lagrangian, 4, 12
- landmark, 21
- Laplacian
  - eigenmaps, 8, 11–14, 21, 23
  - matrix, 2, 3
  - normalized, 12–14
  - unnormalized, 12
- learning
  - manifold, 3
  - rate, 16
- likelihood, 16
- LLE, 14, 21, 23
- locally linear, 13
- majorization, 7
- manifold, 2, 3, 7–9, 15, 17
  - disconnected, 10
  - learning, 3
  - non-convex, 9
  - smooth, 3, 9, 14
  - submanifold, 2
  - underlying, 20
  - unfolding, 7, 8, 10, 11
- map
  - self-organizing, 2, 8, 15, 16
- mapping
  - nonlinear, 6, 9, 23

- Sammon's nonlinear mapping, 3
- topographic, 16
- topographic mapping, 2
- mass, 8, 16, 17
- MDS, *see* multidimensional scaling, 3, 9, 20
- Mercer kernel, 13
- min-cut, 12
- MNIST, 20
- Moore-Penrose, 15
- multidimensional scaling, 3, 19
  - metric
    - classical, 2, 5
- MVU, *see* variance, 11, 21
- neighborhood, 7, 11, 19
  - $K$ -ary, 2, 7, 9, 10, 14, 19, 21
  - $\epsilon$ -ball, 2, 7, 10, 14
  - graph, 10, 11, 13–15, 17, 18
  - radius, 16, 17
  - relationship, 7, 18
- network
  - electrical, 13
  - neural, 3, 16
- NLDR, *see* see dimensionality, 3, 7, 13, 20
- NLM, *see* Sammon, 6, 9, 21
- optimization, 7, 23
  - constrained, 11
  - convex, 13
  - heuristic, 15–18, 23
  - non-spectral, 23
  - pseudo-Newton, 23
  - spectral, 23
- orthogonal, 4, 5
- parameter, 21
- path, 10, 11
  - shortest, 9, 18, 23
- PCA, *see* component, 3, 5, 6, 15, 22
  - kernel, 13
- permutation
  - matrix, 19
- pixel, 20
- placement
  - force-directed, 8
  - graph, 8
- principal component analysis, *see* component
- projection, 7, 15, 16
- proximity, 2, 8
- pseudo-inverse, 15
- quality, 18
  - assessment, 18, 21, 22
  - criterion, 18, 19
- random walk, 2, 13
- rank, 2, 14, 18
  - ex aequo, 18
  - nonreflexive, 18
  - quality criteria, 2
- reconstruction, 3–5, 15
  - coefficient, 14
- regularization, 14
- representation
  - discrete, 9, 15
  - two-dimensional, 20
- resistance, 13
- Sammon, *see* mapping, 6, 9, 18, 21, 23
- scatterplot, 19
- semidefinite programming, 11, 22
- Shepard
  - diagram, 19
- similarity, 2, 3, 8, 11, 23
  - matrix, 22
    - sparse, 8, 13, 14, 23
  - preservation, 23
- singular value, 5
- SOM, 15–18, 21, 23
- space
  - Euclidean, 9
  - feature, 13
  - high-dimensional, 1, 2, 8, 9, 15, 17–19
  - latent, 7, 21
  - low-dimensional, 1
  - visualization, 19

sparse, 12–14, 22, 23

sparsity, 7

spiral, 7, 19

spring, 8, 16, 17

SSTRESS, 7

step

    function, 16

    size, 17

STRAIN, 6, 7

STRESS, 6, 7

stress, 6, 18

    squared, 7

subspace, 4, 5

    linear, 3, 6, 7, 15, 16

    nonlinear, 8, 15

Swiss roll, 7, 8, 19, 21

topology, 3

unfolding

    manifold, 7

    maximum variance, 8

variance, 22

    maximum variance unfolding, 8, 10, 13

    preservation, 5, 20

vector quantization, 16, 23

vicinity, 8

visualization, 2, 15, 19, 22

    space, 15

weight, 6, 9, 12, 13, 15

XOM, 8, 18