To appear as a part of an upcoming academic book on dimensionality reduction and manifold learning.

Multidimensional Scaling, Sammon Mapping, and Isomap: Tutorial and Survey

Benyamin Ghojogh BGHOJOGH@UWATERLOO.CA
Department of Electrical and Computer Engineering,
Machine Learning Laboratory, University of Waterloo, Waterloo, ON, Canada

Ali Ghodsi ALI.GHODSI@UWATERLOO.CA
Department of Statistics and Actuarial Science & David R. Cheriton School of Computer Science,
Data Analytics Laboratory, University of Waterloo, Waterloo, ON, Canada

Fakhri Karray KARRAY@UWATERLOO.CA
Department of Electrical and Computer Engineering,
Centre for Pattern Analysis and Machine Intelligence, University of Waterloo, Waterloo, ON, Canada

Mark Crowley MCROWLEY@UWATERLOO.CA
Department of Electrical and Computer Engineering,
Machine Learning Laboratory, University of Waterloo, Waterloo, ON, Canada

Abstract

Multidimensional Scaling (MDS) is one of the first fundamental manifold learning methods. It can be categorized into several methods, i.e., classical MDS, kernel classical MDS, metric MDS, and non-metric MDS. Sammon mapping and Isomap can be considered as special cases of metric MDS and kernel classical MDS, respectively. In this tutorial and survey paper, we review the theory of MDS, Sammon mapping, and Isomap in detail. We explain all the mentioned categories of MDS. Then, Sammon mapping, Isomap, and kernel Isomap are explained. Out-of-sample embedding for MDS and Isomap using eigenfunctions and kernel mapping are introduced. Then, Nystrom approximation and its use in landmark MDS and landmark Isomap are introduced for big data embedding. We also provide some simulations for illustrating the embedding by these methods.

1. Introduction

Multidimensional Scaling (MDS) (Cox & Cox, 2008), first proposed in (Torgerson, 1952), is one of the earliest proposed manifold learning methods. It can be used for manifold learning, dimensionality reduction, and feature extraction (Ghojogh et al., 2019c). The idea of MDS is to preserve the similarity (Torgerson, 1965) or dissimilarity/distances (Beals et al., 1968) of points in the low-dimensional embedding space. Hence, it fits the data locally to capture the global structure of data (Saul & Roweis, 2003). MDS can be categorized into classical MDS, metric MDS, and non-metric MDS.

In later approaches, Sammon mapping (Sammon, 1969) was proposed which is a special case of the distance-based metric MDS. One can consider Sammon mapping as the first proposed nonlinear manifold learning method (Ghojogh et al., 2019b). The disadvantage of Sammon mapping is its iterative solution of optimization, which makes this method a little slow.

The classical MDS can be generalized to have kernel classical MDS in which any valid kernel can be used. Isomap (Tenenbaum et al., 2000) is a special case of the kernel classical MDS which uses a kernel constructed from geodesic distances between points. Because of the nonlinearity of geodesic distance, Isomap is also a nonlinear manifold learning method.

MDS and its special cases, Sammon mapping, and Isomap have had different applications (Young, 2013). For example, MDS has been used for facial expression recognition (Russell & Bullock, 1985; Katsikitis, 1997). Kernel Isomap has also been used for this application (Zhao & Zhang, 2011).

The goal is to embed the high-dimensional input data

$\{\boldsymbol{x}_i\}_{i=1}^n$ into the lower dimensional embedded data $\{\boldsymbol{y}_i\}_{i=1}^n$ where n is the number of data points. We denote the dimensionality of input and embedding spaces by d and $p \leq d$, respectively, i.e. $\boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{y}_i \in \mathbb{R}^p$. We denote $\mathbb{R}^{d\times n} \ni \boldsymbol{X} := [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]$ and $\mathbb{R}^{p\times n} \ni \boldsymbol{Y} := [\boldsymbol{y}_1, \dots, \boldsymbol{y}_n]$.

The remainder of this paper is organized as follows. Section 2 explains MDS and its different categories, i.e., classical MDS, generalized classical MDS (kernel classical MDS), metric MDS, and non-metric MDS. Sammon mapping and Isomap are introduced in Sections 3 and 4, respectively. Section 5 introduced the methods for out-of-sample extensions of MDS and Isomap methods. Landmark MDS and landmark Isomap, for big data embedding, are explained in Section 6. Some simulations for illustrating the results of embedding are provided in Section 7. Finally, Section 8 concludes the paper.

2. Multidimensional Scaling

MDS, first proposed in (Torgerson, 1952), can be divided into several different categories (Cox & Cox, 2008; Borg & Groenen, 2005), i.e., classical MDS, metric MDS, and non-metric MDS. Note that the results of these are different (Jung, 2013). In the following, we explain all three categories.

2.1. Classical Multidimensional Scaling

2.1.1. Classical MDS with Euclidean Distance

The *classical MDS* is also referred to as *Principal Coordinates Analysis (PCoA)*, or *Torgerson Scaling*, or *TorgersonGower scaling* (Gower, 1966). The goal of classical MDS is to preserve the similarity of data points in the embedding space as it was in the input space (Torgerson, 1965). One way to measure similarity is inner product. Hence, we can minimize the difference of similarities in the input and embedding spaces:

$$\underset{\{\boldsymbol{y}_i\}_{i=1}^n}{\text{minimize}} \quad c_1 := \sum_{i=1}^n \sum_{j=1}^n (\boldsymbol{x}_i^\top \boldsymbol{x}_j - \boldsymbol{y}_i^\top \boldsymbol{y}_j)^2, \tag{1}$$

whose matrix form is:

$$\underset{\boldsymbol{Y}}{\text{minimize}} \quad c_1 = ||\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y}||_F^2, \tag{2}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, and $\boldsymbol{X}^\top \boldsymbol{X}$ and $\boldsymbol{Y}^\top \boldsymbol{Y}$ are the Gram matrices of the original data $\boldsymbol{X}$ and the embedded data $\boldsymbol{Y}$, respectively.

The objective function, in Eq. (2), is simplified as:

$$\begin{aligned}
&||\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y}||_F^2 \\
&\quad = \textbf{tr}\big[(\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y})^\top (\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y})\big] \\
&\quad = \textbf{tr}\big[(\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y})(\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y})\big] \\
&\quad = \textbf{tr}\big[(\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y})^2\big],
\end{aligned}$$

where $\textbf{tr}(.)$ denotes the trace of matrix. If we decompose $\boldsymbol{X}^\top \boldsymbol{X}$ and $\boldsymbol{Y}^\top \boldsymbol{Y}$ using eigenvalue decomposition (Ghojogh et al., 2019a), we have:

$$\boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{V} \boldsymbol{\Delta} \boldsymbol{V}^\top, \tag{3}$$

$$\boldsymbol{Y}^\top \boldsymbol{Y} = \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top, \tag{4}$$

where eigenvectors are sorted from leading (largest eigenvalue) to trailing (smallest eigenvalue). Note that, rather than eigenvalue decomposition of $\boldsymbol{X}^\top \boldsymbol{X}$ and $\boldsymbol{Y}^\top \boldsymbol{Y}$, one can decompose $\boldsymbol{X}$ and $\boldsymbol{Y}$ using Singular Value Decomposition (SVD) and take the right singular vectors of $\boldsymbol{X}$ and $\boldsymbol{Y}$ as $\boldsymbol{V}$ and $\boldsymbol{Q}$, respectively. The matrices $\boldsymbol{\Delta}$ and $\boldsymbol{\Psi}$ are the obtained by squaring the singular values (to power 2). See (Ghojogh & Crowley, 2019, Proposition 1) for proof.

The objective function can be further simplified as:

$$\begin{aligned}
\therefore \quad & ||\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y}||_F^2 \\
&= \textbf{tr}\big[(\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y})^2\big] \\
&= \textbf{tr}\big[(\boldsymbol{V} \boldsymbol{\Delta} \boldsymbol{V}^\top - \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top)^2\big] \\
&\overset{(a)}{=} \textbf{tr}\big[(\boldsymbol{V} \boldsymbol{\Delta} \boldsymbol{V}^\top - \boldsymbol{V}\boldsymbol{V}^\top \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top \boldsymbol{V}\boldsymbol{V}^\top)^2\big] \\
&= \textbf{tr}\Big[\big(\boldsymbol{V}(\boldsymbol{\Delta} - \boldsymbol{V}^\top \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top \boldsymbol{V})\boldsymbol{V}^\top\big)^2\Big] \\
&= \textbf{tr}\Big[\boldsymbol{V}^2(\boldsymbol{\Delta} - \boldsymbol{V}^\top \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top \boldsymbol{V})^2(\boldsymbol{V}^\top)^2\Big] \\
&\overset{(b)}{=} \textbf{tr}\Big[(\boldsymbol{V}^\top)^2\boldsymbol{V}^2(\boldsymbol{\Delta} - \boldsymbol{V}^\top \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top \boldsymbol{V})^2\Big] \\
&= \textbf{tr}\Big[(\underbrace{\boldsymbol{V}^\top \boldsymbol{V}}_{\boldsymbol{I}})^2(\boldsymbol{\Delta} - \boldsymbol{V}^\top \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top \boldsymbol{V})^2\Big] \\
&\overset{(c)}{=} \textbf{tr}\Big[(\boldsymbol{\Delta} - \boldsymbol{V}^\top \boldsymbol{Q} \boldsymbol{\Psi} \boldsymbol{Q}^\top \boldsymbol{V})^2\Big],
\end{aligned}$$

where (a) and (c) are for $\boldsymbol{V}^\top \boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^\top = \boldsymbol{I}$ because $\boldsymbol{V}$ is a non-truncated (square) orthogonal matrix (where $\boldsymbol{I}$ denotes the identity matrix). The reason of (b) is the cyclic property of trace.

Let $\mathbb{R}^{n\times n} \ni \boldsymbol{M} := \boldsymbol{V}^\top \boldsymbol{Q}$, so:

$$||\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y}||_F^2 = \textbf{tr}\Big[(\boldsymbol{\Delta} - \boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top)^2\Big].$$

Therefore:

$$\begin{aligned}
\therefore \quad & \underset{\boldsymbol{Y}}{\text{minimize}} \ ||\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{Y}^\top \boldsymbol{Y}||_F^2 \\
& \equiv \underset{\boldsymbol{M}, \boldsymbol{\Psi}}{\text{minimize}} \ \textbf{tr}\Big[(\boldsymbol{\Delta} - \boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top)^2\Big].
\end{aligned}$$

The objective function is:

$$\begin{aligned}
c_1 &= \textbf{tr}\Big[(\boldsymbol{\Delta} - \boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top)^2\Big] \\
&= \textbf{tr}(\boldsymbol{\Delta}^2 + (\boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top)^2 - 2\boldsymbol{\Delta} \boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top) \\
&= \textbf{tr}(\boldsymbol{\Delta}^2) + \textbf{tr}((\boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top)^2) - 2\,\textbf{tr}(\boldsymbol{\Delta} \boldsymbol{M} \boldsymbol{\Psi} \boldsymbol{M}^\top).
\end{aligned}$$

As the optimization problem is unconstrained and the objective function is the trace of a quadratic function, the minimum is non-negative.

If we take derivative with respect to the first objective variable, i.e., $\boldsymbol{M}$, we have:

$$\begin{aligned}\mathbb{R}^{n\times n} \ni \frac{\partial c_1}{\partial \boldsymbol{M}} &= 2(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)\boldsymbol{M}\boldsymbol{\Psi} - 2\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi} \overset{\text{set}}{=} \boldsymbol{0} \\ &\implies (\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)(\boldsymbol{M}\boldsymbol{\Psi}) = (\boldsymbol{\Delta})(\boldsymbol{M}\boldsymbol{\Psi}) \\ &\overset{(a)}{\implies} \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top = \boldsymbol{\Delta}, \end{aligned} \tag{5}$$

where (a) is because $\boldsymbol{M}\boldsymbol{\Psi} \neq \boldsymbol{0}$.

For the derivative with respect to the second objective variable, i.e., $\boldsymbol{\Psi}$, we simplify the objective function a little bit:

$$\begin{aligned} c_1 &= \textbf{tr}(\boldsymbol{\Delta}^2) + \textbf{tr}((\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2) - 2\,\textbf{tr}(\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top) \\ &= \textbf{tr}(\boldsymbol{\Delta}^2) + \textbf{tr}(\boldsymbol{M}^2\boldsymbol{\Psi}^2\boldsymbol{M}^{\top 2}) - 2\,\textbf{tr}(\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top) \\ &\overset{(a)}{=} \textbf{tr}(\boldsymbol{\Delta}^2) + \textbf{tr}(\boldsymbol{M}^{\top 2}\boldsymbol{M}^2\boldsymbol{\Psi}^2) - 2\,\textbf{tr}(\boldsymbol{M}^\top\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}) \\ &= \textbf{tr}(\boldsymbol{\Delta}^2) + \textbf{tr}((\boldsymbol{M}^\top\boldsymbol{M}\boldsymbol{\Psi})^2) - 2\,\textbf{tr}(\boldsymbol{M}^\top\boldsymbol{\Delta}\boldsymbol{M}\boldsymbol{\Psi}), \end{aligned}$$

where (a) is because of the cyclic property of trace.

Taking derivative with respect to the second objective variable, i.e., $\boldsymbol{\Psi}$, gives:

$$\begin{aligned}\mathbb{R}^{n\times n} \ni \frac{\partial c_1}{\partial \boldsymbol{\Psi}} &= 2\boldsymbol{M}^\top(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)\boldsymbol{M} - 2\boldsymbol{M}^\top\boldsymbol{\Delta}\boldsymbol{M} \overset{\text{set}}{=} \boldsymbol{0} \\ &\implies \boldsymbol{M}^\top(\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)\boldsymbol{M} = \boldsymbol{M}^\top(\boldsymbol{\Delta})\boldsymbol{M} \\ &\overset{(a)}{\implies} \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top = \boldsymbol{\Delta}, \end{aligned} \tag{6}$$

where (a) is because $\boldsymbol{M} \neq \boldsymbol{0}$. Both Eqs. (5) and (6) are:

$$\boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top = \boldsymbol{\Delta},$$

whose one possible solution is:

$$\boldsymbol{M} = \boldsymbol{I}, \tag{7}$$
$$\boldsymbol{\Psi} = \boldsymbol{\Delta}. \tag{8}$$

which means that the minimum value of the non-negative objective function $\textbf{tr}((\boldsymbol{\Delta} - \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{M}^\top)^2)$ is zero.

We had $\boldsymbol{M} = \boldsymbol{V}^\top\boldsymbol{Q}$. Therefore, according to Eq. (7), we have:

$$\therefore \quad \boldsymbol{V}^\top\boldsymbol{Q} = \boldsymbol{I} \implies \boldsymbol{Q} = \boldsymbol{V}. \tag{9}$$

According to Eq. (4), we have:

$$\begin{aligned}\boldsymbol{Y}^\top\boldsymbol{Y} = \boldsymbol{Q}\boldsymbol{\Psi}\boldsymbol{Q}^\top &\overset{(a)}{=} \boldsymbol{Q}\boldsymbol{\Psi}^{\frac{1}{2}}\boldsymbol{\Psi}^{\frac{1}{2}}\boldsymbol{Q}^\top \implies \boldsymbol{Y} = \boldsymbol{\Psi}^{\frac{1}{2}}\boldsymbol{Q}^\top \\ &\overset{(8),(9)}{\implies} \boldsymbol{Y} = \boldsymbol{\Delta}^{\frac{1}{2}}\boldsymbol{V}^\top, \end{aligned} \tag{10}$$

where (a) can be done because $\boldsymbol{\Psi}$ does not include negative entry as the gram matrix $\boldsymbol{Y}^\top\boldsymbol{Y}$ is positive semi-definite by definition.

In summary, for embedding $\boldsymbol{X}$ using classical MDS, the eigenvalue decomposition of $\boldsymbol{X}^\top\boldsymbol{X}$ is obtained as in Eq. (3). Then, using Eq. (10), $\boldsymbol{Y} \in \mathbb{R}^{n\times n}$ is obtained. Truncating this $\boldsymbol{Y}$ to have $\boldsymbol{Y} \in \mathbb{R}^{p\times n}$, with the first (top) p rows, gives us the p-dimensional embedding of the n points. Note that the leading p columns are used because singular values are sorted from largest to smallest in SVD which can be used for Eq. (3).

2.1.2. Generalized Classical MDS (Kernel Classical MDS)

If $d_{ij}^2 = ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2$ is the squared Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we have:

$$\begin{aligned} d_{ij}^2 &= ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2 = (\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j) \\ &= \boldsymbol{x}_i^\top\boldsymbol{x}_i - \boldsymbol{x}_i^\top\boldsymbol{x}_j - \boldsymbol{x}_j^\top\boldsymbol{x}_i + \boldsymbol{x}_j^\top\boldsymbol{x}_j \\ &= \boldsymbol{x}_i^\top\boldsymbol{x}_i - 2\boldsymbol{x}_i^\top\boldsymbol{x}_j + \boldsymbol{x}_j^\top\boldsymbol{x}_j = \boldsymbol{G}_{ii} - 2\boldsymbol{G}_{ij} + \boldsymbol{G}_{jj}, \end{aligned}$$

where $\mathbb{R}^{n\times n} \ni \boldsymbol{G} := \boldsymbol{X}^\top\boldsymbol{X}$ is the Gram matrix. If $\mathbb{R}^n \ni \boldsymbol{g} := [\boldsymbol{g}_1, \dots, \boldsymbol{g}_n] = [\boldsymbol{G}_{11}, \dots, \boldsymbol{G}_{nn}] = \textbf{diag}(\boldsymbol{G})$, we have:

$$\begin{aligned} d_{ij}^2 &= \boldsymbol{g}_i - 2\boldsymbol{G}_{ij} + \boldsymbol{g}_j, \\ \boldsymbol{D} &= \boldsymbol{g}\boldsymbol{1}^\top - 2\boldsymbol{G} + \boldsymbol{1}\boldsymbol{g}^\top = \boldsymbol{1}\boldsymbol{g}^\top - 2\boldsymbol{G} + \boldsymbol{g}\boldsymbol{1}^\top, \end{aligned}$$

where $\boldsymbol{1}$ is the vector of ones and $\boldsymbol{D}$ is the distance matrix with squared Euclidean distance (d_{ij}^2 as its elements). Let $\mathbb{R}^{n\times n} \ni \boldsymbol{H} := \boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top$ denote the centering matrix. We double-center the matrix $\boldsymbol{D}$ as follows (Oldford, 2018):

$$\begin{aligned} \boldsymbol{H}\boldsymbol{D}\boldsymbol{H} &= (\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{D}(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top) \\ &= (\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)(\boldsymbol{1}\boldsymbol{g}^\top - 2\boldsymbol{G} + \boldsymbol{g}\boldsymbol{1}^\top)(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top) \\ &= \big[\underbrace{(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{1}}_{=\boldsymbol{0}}\boldsymbol{g}^\top - 2(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{G} \\ &\quad + (\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{g}\boldsymbol{1}^\top\big](\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top) \\ &= -2(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{G}(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top) \\ &\quad + (\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{g}\underbrace{\boldsymbol{1}^\top(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)}_{=\boldsymbol{0}} \\ &= -2(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{G}(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top) = -2\,\boldsymbol{H}\boldsymbol{G}\boldsymbol{H} \end{aligned}$$

$$\therefore \quad \boldsymbol{H}\boldsymbol{G}\boldsymbol{H} = \boldsymbol{H}\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{H} = -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H}. \tag{11}$$

Note that $(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top)\boldsymbol{1} = \boldsymbol{0}$ and $\boldsymbol{1}^\top(\boldsymbol{I} - \frac{1}{n}\boldsymbol{1}\boldsymbol{1}^\top) = \boldsymbol{0}$ because removing the row mean of $\boldsymbol{1}$ and column mean of of $\boldsymbol{1}^\top$ results in the zero vectors, respectively.

If data $\boldsymbol{X}$ are already centered, i.e., the mean has been removed ($\boldsymbol{X} \leftarrow \boldsymbol{X}\boldsymbol{H}$), Eq. (11) becomes:

$$\boldsymbol{X}^\top \boldsymbol{X} = -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H}. \tag{12}$$

Corollary 1. *If using Eq. (3) as Gram matrix, the classical MDS uses the Euclidean distance as its metric. Because of using Euclidean distance, the classical MDS using Gram matrix is a* ***linear*** *subspace learning method.*

Proof. The Eq. (3) in classical MDS is the eigenvalue decomposition of the Gram matrix $\boldsymbol{X}^\top \boldsymbol{X}$. According to Eq. (12), this Gram matrix can be restated to an expression based on squared Euclidean distance. Hence, the classical MDS with Eq. (3) uses Euclidean distance and is linear, consequently. □

In Eq. (11) or (12), we can write a general kernel matrix (Hofmann et al., 2008) rather than the double-centered Gram matrix, to have (Cox & Cox, 2008):

$$\mathbb{R}^{n\times n} \ni \boldsymbol{K} = -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H}. \tag{13}$$

Note that the classical MDS with Eq. (3) is using a linear kernel $\boldsymbol{X}^\top \boldsymbol{X}$ for its kernel. This is another reason for why classical MDS with Eq. (3) is a linear method. It is also noteworthy that Eq. (13) can be used for unifying the spectral dimensionality reduction methods as special cases of kernel principal component analysis with different kernels. See (Ham et al., 2004; Bengio et al., 2004a) and (Strange & Zwiggelaar, 2014, Table 2.1) for more details.

Comparing Eqs. (11), (12), and (13) with Eq. (3) shows that we can use a general kernel matrix, like Radial Basis Function (RBF) kernel, in classical MDS to have *generalized classical MDS*. In summary, for embedding $\boldsymbol{X}$ using classical MDS, the eigenvalue decomposition of the kernel matrix $\boldsymbol{K}$ is obtained similar to Eq. (3):

$$\boldsymbol{K} = \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^\top. \tag{14}$$

Then, using Eq. (10), $\boldsymbol{Y} \in \mathbb{R}^{n\times n}$ is obtained. It is noteworthy that in this case, we are replacing $\boldsymbol{X}^\top \boldsymbol{X}$ with the kernel $\boldsymbol{K} = \boldsymbol{\Phi}(\boldsymbol{X})^\top \boldsymbol{\Phi}(\boldsymbol{X})$ and then, according to Eqs. (10) and (14), we have:

$$\boldsymbol{K} = \boldsymbol{Y}^\top \boldsymbol{Y}. \tag{15}$$

Truncating the $\boldsymbol{Y}$, obtained from Eq. (10), to have $\boldsymbol{Y} \in \mathbb{R}^{p\times n}$, with the first (top) p rows, gives us the p-dimensional embedding of the n points. It is noteworthy that, because of using kernel in the generalized classical MDS, one can name it the *kernel classical MDS*.

2.1.3. Equivalence of PCA and kernel PCA with Classical MDS and Generalized Classical MDS, Respectively

Proposition 1. *Classical MDS with Euclidean distance is equivalent to Principal Component Analysis (PCA). Moreover, the generalized classical MDS is equivalent to kernel PCA.*

Proof. On one hand, the Eq. (3) can be obtained by the SVD of $\boldsymbol{X}$. The projected data onto classical MDS subspace is obtained by Eq. (10) which is $\boldsymbol{\Delta}\boldsymbol{V}^\top$. On the other hand, according to (Ghojogh & Crowley, 2019, Eq. 42), the projected data onto PCA subspace is $\boldsymbol{\Delta}\boldsymbol{V}^\top$ where $\boldsymbol{\Delta}$ and $\boldsymbol{V}^\top$ are from the SVD of $\boldsymbol{X}$. Comparing these shows that classical MDS is equivalent to PCA.

Moreover, Eq. (14) is the eigenvalue decomposition of the kernel matrix. The projected data onto the generalized classical MDS subspace is obtained by Eq. (10) which is $\boldsymbol{\Delta}\boldsymbol{V}^\top$. According to (Ghojogh & Crowley, 2019, Eq. 62), the projected data onto the kernel PCA subspace is $\boldsymbol{\Delta}\boldsymbol{V}^\top$ where $\boldsymbol{\Delta}$ and $\boldsymbol{V}^\top$ are from the eigenvalue decomposition of the kernel matrix; see (Ghojogh & Crowley, 2019, Eq. 61). Comparing these shows that the generalized classical MDS is equivalent to kernel PCA. □

2.2. Metric Multidimensional Scaling

Recall that the classical MDS tries to preserve the similarities of points in the embedding space. In later approaches after classical MDS, the cost function was changed to preserve the distances rather than the similarities (Lee & Verleysen, 2007; Bunte et al., 2012). *Metric MDS* has this opposite view and tries to preserve the distances of points in the embedding space (Beals et al., 1968). For this, it minimizes the difference of distances of points in the input and embedding spaces (Ghodsi, 2006). The cost function in metric MDS is usually referred to as the *stress function* (Mardia, 1978; De Leeuw, 2011). This method is named metric MDS because it uses distance metric in its optimization. The optimization in metric MDS is:

$$\begin{aligned}&\underset{\{\boldsymbol{y}_i\}_{i=1}^n}{\text{minimize}} \\ &c_2 := \left(\frac{\sum_{i=1}^n \sum_{j=1,j<i}^n \big(d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)\big)^2}{\sum_{i=1}^n \sum_{j=1,j<i}^n d_x^2(\boldsymbol{x}_i, \boldsymbol{x}_j)}\right)^{\frac{1}{2}},\end{aligned} \tag{16}$$

or, without the normalization factor:

$$\begin{aligned}&\underset{\{\boldsymbol{y}_i\}_{i=1}^n}{\text{minimize}} \\ &c_2 := \left(\sum_{i=1}^n \sum_{j=1,j<i}^n \big(d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)\big)^2\right)^{\frac{1}{2}},\end{aligned} \tag{17}$$

where $d_x(.,.)$ and $d_y(.,.)$ denote the distance metrics in the input and the embedded spaces, respectively.

The Eqs. (16) and (17) use indices $j < i$ rather than $j \neq i$ because the distance metric is symmetric and it is not necessary to consider the distance of the j-th point from the i-th point when we already have considered the distance of the i-th point from the j-th point. Note that in Eq. (16) and (17), d_y is usually the Euclidean distance, i.e. $d_y = \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2$, while d_x can be any valid metric distance such as the Euclidean distance.

The optimization problem (16) can be solved using either gradient descent or Newton's method. Note that the classical MDS is a linear method and has a closed-form solution; however, the metric and non-metric MDS methods are **nonlinear** but do *not have closed-form solutions* and should be solved iteratively. Note that in mathematics, whenever you get something, you lose something. Likewise, here, the method has become nonlinear but lost its closed form solution and became iterative.

Inspired by (Sammon, 1969), we can use diagonal quasi-Newton's method for solving this optimization problem. If we consider the vectors component-wise, the diagonal quasi-Newton's method updates the solution as (Lee & Verleysen, 2007):

$$y_{i,k}^{(\nu+1)} := y_{i,k}^{(\nu)} - \eta \left| \frac{\partial^2 c_2}{\partial y_{i,k}^2} \right|^{-1} \frac{\partial c_2}{\partial y_{i,k}}, \tag{18}$$

where η is the learning rate, $y_{i,k}$ is the k-th element of the i-th embedded point $\mathbb{R}^p \ni \boldsymbol{y}_i = [y_{i,1}, \dots, y_{i,p}]^\top$, and $|\cdot|$ is the absolute value guaranteeing that we move toward the minimum and not maximum in the Newton's method. If using gradient descent for solving the optimization, we update the solution as:

$$y_{i,k}^{(\nu+1)} := y_{i,k}^{(\nu)} - \eta \frac{\partial c_2}{\partial y_{i,k}}. \tag{19}$$

2.3. Non-Metric Multidimensional Scaling

In *non-metric MDS*, rather than using a distance metric, $d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for the distances between points in the embedding space, we use $f(d_y(\boldsymbol{x}_i, \boldsymbol{x}_j))$ where $f(.)$ is a non-parametric monotonic function. In other words, only the order of dissimilarities is important rather than the amount of dissimilarities (Agarwal et al., 2007; Jung, 2013):

$$\begin{aligned} & d_y(\boldsymbol{y}_i, \boldsymbol{y}_j) \leq d_y(\boldsymbol{y}_k, \boldsymbol{y}_\ell) \iff \\ & \qquad f(d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)) \leq f(d_y(\boldsymbol{x}_k, \boldsymbol{y}_\ell)). \end{aligned} \tag{20}$$

The optimization in non-metric MDS is (Agarwal et al., 2007):

$$\underset{\{\boldsymbol{y}_i\}_{i=1}^n}{\text{minimize}} \quad c_3 := \left(\frac{\sum_{i=1}^n \sum_{j=1, j<i}^n \big(d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - f(d_y(\boldsymbol{y}_i, \boldsymbol{y}_j))\big)^2}{\sum_{i=1}^n \sum_{j=1, j<i}^n d_x^2(\boldsymbol{x}_i, \boldsymbol{x}_j)} \right)^{\frac{1}{2}}. \tag{21}$$

An examples of non-metric MDS is Smallest Space Analysis (Schlesinger & Guttman, 1969). Another example is Kruskal's non-metric MDS or Shepard-Kruskal Scaling (SKS) (Kruskal, 1964a;b). In Kruskal's non-metric MDS, the function $f(.)$ is the regression, where $f(d_y(\boldsymbol{y}_i, \boldsymbol{y}_j))$ is predicted from regression which preserves the order of dissimilarities (Holland, 2008; Agarwal et al., 2007). The Eq. (21) with $f(.)$ as the regression function, which is used in Kruskal's non-metric MDS, is called *Stress-1 formula* (Agarwal et al., 2007; Holland, 2008; Jung, 2013).

3. Sammon Mapping

Sammon mapping (Sammon, 1969) is a special case of metric MDS; hence, it is a **nonlinear** method. It is probably correct to call this method the first proposed nonlinear method for manifold learning (Ghojogh et al., 2019b).

This method has different names in the literature such as *Sammon's nonlinear mapping*, *Sammon mapping*, and *Nonlinear Mapping (NLM)* (Lee & Verleysen, 2007). Sammon originally named it NLM (Sammon, 1969). Its most well-known name is Sammon mapping.

The optimization problem in Sammon mapping is almost a weighted version of Eq. (16), formulated as:

$$\underset{\{\boldsymbol{y}_i\}_{i=1}^n}{\text{minimize}} \quad \frac{1}{a} \sum_{i=1}^n \sum_{j=1, j<i}^n w_{ij} \big(d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)\big)^2, \tag{22}$$

where w_{ij} is the weight and a is the normalizing factor. The $d_x(.,.)$ can be any metric but usually is considered to be Euclidean distance for simplicity (Lee & Verleysen, 2007). The $d_y(.,.)$, however, is Euclidean distance metric.

In Sammon mapping, the weights and the normalizing factor in Eq. (22) are:

$$w_{ij} = \frac{1}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)}, \tag{23}$$

$$a = \sum_{i=1}^n \sum_{j=1, j<i}^n d_x(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{24}$$

The weight w_{ij} in Sammon mapping is giving more credit to the small distances (neighbor points) focusing on preserving the "local" structure of the manifold; hence it fits the manifold locally (Saul & Roweis, 2003).

Substituting Eqs. (23) and (24) in Eq. (22) gives:

$$\begin{aligned} \underset{\boldsymbol{Y}}{\text{minimize}} \quad & c_4 := \frac{1}{\sum_{i=1}^n \sum_{j=1, j<i}^n d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)} \times \\ & \sum_{i=1}^n \sum_{j=1, j<i}^n \frac{\big(d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)\big)^2}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)}. \end{aligned} \tag{25}$$

Sammon used diagonal quasi-Newton's method for solving this optimization problem (Sammon, 1969). Hence, Eq.

(18) is utilized. The learning rate η is named the *magic factor* in (Sammon, 1969). For solving optimization, both gradient and second derivative are required. In the following, we derive these two.

Note that, in practice, the classical MDS or PCA is used for initialization of points in Sammon mapping optimization.

Proposition 2. *The gradient of the cost function c with respect to $y_{i,k}$ is (Sammon, 1969; Lee & Verleysen, 2007):*

$$\frac{\partial c_4}{\partial y_{i,k}} = \frac{-2}{a} \sum_{i=1}^{n} \sum_{j=1, j<i}^{n} \frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} (y_{i,k} - y_{j,k}). \tag{26}$$

Proof. Proof is according to (Lee & Verleysen, 2007). According to chain rule, we have:

$$\frac{\partial c_4}{\partial y_{i,k}} = \frac{\partial c_4}{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)} \times \frac{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial y_{i,k}}.$$

The first derivative is:

$$\frac{\partial c_4}{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)} = \frac{-2}{a} \sum_{i=1}^{n} \sum_{j=1, j<i}^{n} \frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)},$$

and using the chain rule, the second derivative is:

$$\frac{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial y_{i,k}} = \frac{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial d_y^2(\boldsymbol{y}_i, \boldsymbol{y}_j)} \times \frac{\partial d_y^2(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial y_{i,k}}.$$

We have:

$$\frac{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial d_y^2(\boldsymbol{y}_i, \boldsymbol{y}_j)} = 1 / \frac{\partial d_y^2(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)} = 1/(2 d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)).$$

Also we have:

$$d_y^2(\boldsymbol{y}_i, \boldsymbol{y}_j) = ||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^2 = \sum_{k=1}^{p} (y_{i,k} - y_{j,k})^2.$$

Therefore:

$$\frac{\partial d_y^2(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial y_{i,k}} = 2\,(y_{i,k} - y_{j,k}),$$

Therefore:

$$\therefore \quad \frac{\partial d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\partial y_{i,k}} = \frac{y_{i,k} - y_{j,k}}{d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}. \tag{27}$$

Finally, we have:

$$\therefore \quad \frac{\partial c_4}{\partial y_{i,k}} = \frac{-2}{a} \sum_{i=1}^{n} \sum_{j=1, j<i}^{n} \frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} (y_{i,k} - y_{j,k}),$$

which is the gradient mentioned in the proposition. Q.E.D. □

Proposition 3. *The second derivative of the cost function c with respect to $y_{i,k}$ is (Sammon, 1969; Lee & Verleysen, 2007):*

$$\frac{\partial^2 c_4}{\partial y_{i,k}^2} = \frac{-2}{a} \sum_{i=1}^{n} \sum_{j=1, j<i}^{n} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} - \frac{(y_{i,k} - y_{j,k})^2}{d_y^3(\boldsymbol{y}_i, \boldsymbol{y}_j)} \Big). \tag{28}$$

Proof. We have:

$$\frac{\partial^2 c_4}{\partial y_{i,k}^2} = \frac{\partial}{\partial y_{i,k}} \Big(\frac{\partial c_4}{\partial y_{i,k}} \Big),$$

where $\partial c_4 / \partial y_{i,k}$ is Eq. (26). Therefore:

$$\frac{\partial^2 c_4}{\partial y_{i,k}^2} = \frac{-2}{a} \sum_{i=1}^{n} \sum_{j=1, j<i}^{n} \frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} (y_{i,k} - y_{j,k}) \Big).$$

We have:

$$\begin{aligned}
&\frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} (y_{i,k} - y_{j,k}) \Big) \\
&= (y_{i,k} - y_{j,k}) \frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} \Big) \\
&+ \frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} \underbrace{\frac{\partial}{\partial y_{i,k}} (y_{i,k} - y_{j,k})}_{=1}.
\end{aligned}$$

Note that:

$$\begin{aligned}
&\frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} \Big) \\
&= \frac{1}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)} \frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} \Big) \\
&= \frac{1}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)} \frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)}{d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} - 1 \Big) \\
&= \underbrace{\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)}}_{=1} \frac{\partial}{\partial y_{i,k}} \Big(\frac{1}{d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} \Big) - \underbrace{\frac{\partial}{\partial y_{i,k}} (1)}_{=0} \\
&= \frac{-1}{d_y^2(\boldsymbol{x}_i, \boldsymbol{x}_j)} \frac{\partial}{\partial y_{i,k}} (d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)) \\
&\overset{(27)}{=} \frac{-1}{d_y^2(\boldsymbol{x}_i, \boldsymbol{x}_j)} \frac{y_{i,k} - y_{j,k}}{d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}.
\end{aligned}$$

Therefore:

$$\therefore \quad \frac{\partial}{\partial y_{i,k}} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} (y_{i,k} - y_{j,k}) \Big) = \frac{-(y_{i,k} - y_{j,k})^2}{d_y^3(\boldsymbol{y}_i, \boldsymbol{y}_j)} + \frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)}.$$

Therefore:

$$\therefore \quad \frac{\partial^2 c_4}{\partial y_{i,k}^2} = \frac{-2}{a} \sum_{i=1}^{n} \sum_{j=1,j<i}^{n} \Big(\frac{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j)}{d_x(\boldsymbol{x}_i, \boldsymbol{x}_j)\, d_y(\boldsymbol{x}_i, \boldsymbol{x}_j)} - \frac{(y_{i,k} - y_{j,k})^2}{d_y^3(\boldsymbol{y}_i, \boldsymbol{y}_j)} \Big),$$

which is the derivative mentioned in the proposition. Q.E.D. □

It is noteworthy that for better time complexity of the Sammon mapping, one can use the k-Nearest Neighbors (kNN) rather than the whole data (Ghojogh et al., 2020):

$$\underset{\{\boldsymbol{y}_i\}_{i=1}^n}{\text{minimize}} \quad \frac{1}{a} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \big(d_x(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_y(\boldsymbol{y}_i, \boldsymbol{y}_j) \big)^2, \tag{29}$$

where $\mathcal{N}_i$ denotes the set of indices of kNN of the i-th point.

4. Isomap

4.1. Isomap

Isomap (Tenenbaum et al., 2000) is a special case of the generalized classical MDS, explained in Section 2.1.2. Rather than the Euclidean distance, Isomap uses an approximation of the geodesic distance. As was explained, the classical MDS is linear; hence, it cannot capture the nonlinearity of the manifold. Isomap makes use of the geodesic distance to make the generalized classical MDS nonlinear.

4.1.1. Geodesic Distance

The *geodesic distance* is the length of shortest path between two points on the possibly curvy manifold. It is ideal to use the geodesic distance; however, calculation of the geodesic distance is very difficult because it requires traversing from a point to another point on the manifold. This calculation requires differential geometry and Riemannian manifold calculations (Aubin, 2001). Therefore, Isomap approximates the geodesic distance by piece-wise Euclidean distances. It finds the k-Nearest Neighbors (kNN) graph of dataset. Then, the shortest path between two points, through their neighbors, is found using a shortest-path algorithm such as the Dijkstra algorithm or the Floyd-Warshal algorithm (Cormen et al., 2009). A sklearn function in python for this is "graph_shortest_path" from the package "sklearn.utils.graph_shortest_path". Note that the approximated geodesic distance is also refered to as the *curvilinear distance* (Lee et al., 2002). The approximated geodesic distance can be formulated as (Bengio et al., 2004b):

$$\boldsymbol{D}_{ij}^{(g)} := \min_{\boldsymbol{r}} \sum_{i=2}^{l} \|\boldsymbol{r}_i - \boldsymbol{r}_{i+1}\|_2, \tag{30}$$

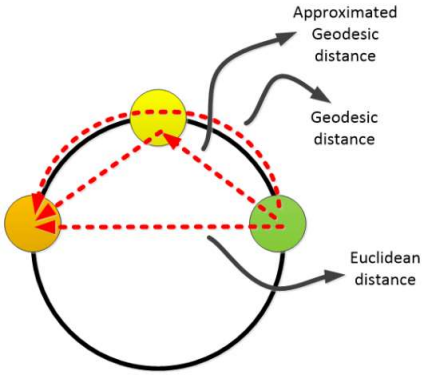

Figure 1. An example of the Euclidean distance, geodesic distance, and approximated geodesic distance using piece-wise Euclidean distances.

where $l \geq 2$ is the length of sequence of points $\boldsymbol{r}_i \in \{\boldsymbol{x}_i\}_{i=1}^n$ and $\boldsymbol{D}_{ij}^{(g)}$ denotes the (i,j)-th element of the geodesic distance matrix $\boldsymbol{D}^{(g)} \in \mathbb{R}^{n \times n}$.

An example of the Euclidean distance, geodesic distance, and the approximated geodesic distance using piece-wise Euclidean distances can be seen in Fig. 1. A real-world example is the distance between Toronto and Athens. The Euclidean distance is to dig the Earth from Toronto to reach Athens directly. The geodesic distance is to move from Toronto to Athens on the curvy Earth by the shortest path between two cities. The approximated geodesic distance is to dig the Earth from Toronto to London in UK, then dig from London to Frankfurt in Germany, then dig from Frankfurt to Rome in Italy, then dig from Rome to Athens. Calculations of lengths of paths in the approximated geodesic distance is much easier than the geodesic distance.

4.1.2. Isomap Formulation

As was mentioned before, Isomap is a special case of the generalized classical MDS with the geodesic distance used. Hence, Isomap uses Eq. (13) as:

$$\mathbb{R}^{n \times n} \ni \boldsymbol{K} = -\frac{1}{2} \boldsymbol{H} \boldsymbol{D}^{(g)} \boldsymbol{H}. \tag{31}$$

It then uses Eqs. (14) and (10) to embed the data. As Isomap uses the nonlinear geodesic distance in its kernel calculation, it is a **nonlinear** method.

4.2. Kernel Isomap

Consider $\boldsymbol{K}(\boldsymbol{D})$ to be Eq. (13). Consequently, we have:

$$\mathbb{R}^{n \times n} \ni \boldsymbol{K}(\boldsymbol{D}^2) = -\frac{1}{2} \boldsymbol{H} \boldsymbol{D}^2 \boldsymbol{H}, \tag{32}$$

where $\boldsymbol{D}$ is the geodesic distance matrix, defined by Eq. (30).

Define the following equation (Cox & Cox, 2008, Section 2.2.8):

$$\mathbb{R}^{n\times n} \ni \boldsymbol{K}' := \boldsymbol{K}(\boldsymbol{D}^2) + 2c\boldsymbol{K}(\boldsymbol{D}) + \frac{1}{2}c^2\boldsymbol{H}. \quad (33)$$

According to (Cailliez, 1983), $\boldsymbol{K}'$ is guaranteed to be positive semi-definite for $c \geq c^*$ where c^* is the largest eigenvalue of the following matrix:

$$\begin{bmatrix} \boldsymbol{0} & 2\boldsymbol{K}(\boldsymbol{D}^2) \\ -\boldsymbol{I} & -4\boldsymbol{K}(\boldsymbol{D}) \end{bmatrix} \in \mathbb{R}^{2n\times 2n}. \quad (34)$$

Kernel Isomap (Choi & Choi, 2004) chooses a value $c \geq c^*$ and uses $\boldsymbol{K}'$ in Eq. (14) and then uses Eq. (10) for embedding the data.

5. Out-of-sample Extensions for MDS and Isomap

So far, we embedded the training dataset $\{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^n$ or $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$ to have their embedding $\{\boldsymbol{y}_i \in \mathbb{R}^p\}_{i=1}^n$ or $\boldsymbol{Y} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_n] \in \mathbb{R}^{p\times n}$. Assume we have some out-of-sample (test data), denoted by $\{\boldsymbol{x}_i^{(t)} \in \mathbb{R}^d\}_{i=1}^{n_t}$ or $\boldsymbol{X}_t = [\boldsymbol{x}_1^{(t)}, \dots, \boldsymbol{x}_n^{(t)}] \in \mathbb{R}^{d\times n_t}$. We want to find their embedding $\{\boldsymbol{y}_i^{(t)} \in \mathbb{R}^p\}_{i=1}^{n_t}$ or $\boldsymbol{Y}_t = [\boldsymbol{y}_1^{(t)}, \dots, \boldsymbol{y}_n^{(t)}] \in \mathbb{R}^{p\times n_t}$ after the training phase.

5.1. Out of Sample for Isomap and MDS Using Eigenfunctions

5.1.1. Eigenfunctions

Consider a Hilbert space $\mathcal{H}_p$ of functions with the inner product $\langle f, g\rangle = \int f(x)g(x)p(x)dx$ with density function $p(x)$. In this space, we can consider the kernel function K_p:

$$(K_p f)(x) = \int K(x, y)\, f(y)\, p(y)\, dy, \quad (35)$$

where the density function can be approximated empirically. The *eigenfunction decomposition* is defined to be (Bengio et al., 2004a;b):

$$(K_p f_k)(x) = \delta'_k f_k(x), \quad (36)$$

where $f_k(x)$ is the k-th *eigenfunction* and δ'_k is the corresponding eigenvalue. If we have the eigenvalue decomposition (Ghojogh et al., 2019a) for the kernel matrix $\boldsymbol{K}$, we have $\boldsymbol{K}\boldsymbol{v}_k = \delta_k \boldsymbol{v}_k$ (see Eq. (14)) where $\boldsymbol{v}_k$ is the k-th eigenvector and δ_k is the corresponding eigenvalue. According to (Bengio et al., 2004b, Proposition 1), we have $\delta'_k = (1/n)\delta_k$.

5.1.2. Embedding Using Eigenfunctions

Proposition 4. *If v_{ki} is the i-th element of the n-dimensional vector $\boldsymbol{v}_k$ and $k(\boldsymbol{x}, \boldsymbol{x}_i)$ is the kernel between vectors $\boldsymbol{x}$ and $\boldsymbol{x}_i$, the eigenfunction for the point $\boldsymbol{x}$ and the i-th training point $\boldsymbol{x}_i$ are:*

$$f_k(\boldsymbol{x}) = \frac{\sqrt{n}}{\delta_k} \sum_{i=1}^n v_{ki}\, \breve{k}_t(\boldsymbol{x}_i, \boldsymbol{x}), \quad (37)$$

$$f_k(\boldsymbol{x}_i) = \sqrt{n}\, v_{ki}, \quad (38)$$

respectively, where $\breve{k}_t(\boldsymbol{x}_i, \boldsymbol{x})$ is the centered kernel between training set and the out-of-sample point $\boldsymbol{x}$.

Let the MDS or Isomap embedding of the point $\boldsymbol{x}$ be $\mathbb{R}^p \ni \boldsymbol{y}(\boldsymbol{x}) = [y_1(\boldsymbol{x}), \dots, y_p(\boldsymbol{x})]^\top$. The k-th dimension of this embedding is:

$$y_k(\boldsymbol{x}) = \sqrt{\delta_k}\, \frac{f_k(\boldsymbol{x})}{\sqrt{n}} = \frac{1}{\sqrt{\delta_k}} \sum_{i=1}^n v_{ki}\, \breve{k}_t(\boldsymbol{x}_i, \boldsymbol{x}). \quad (39)$$

Proof. This proposition is taken from (Bengio et al., 2004b, Proposition 1). For proof, refer to (Bengio et al., 2004a, Proposition 1), (Bengio et al., 2006, Proposition 1), and (Bengio et al., 2003b, Proposition 1 and Theorem 1). More complete proofs can be found in (Bengio et al., 2003a). □

If we have a set of n_t out-of-sample data points, $\breve{k}_t(\boldsymbol{x}_i, \boldsymbol{x})$ is an element of the centered out-of-sample kernel (see (Ghojogh & Crowley, 2019, Appendix C)):

$$\mathbb{R}^{n\times n_t} \ni \breve{\boldsymbol{K}}_t = \boldsymbol{K}_t - \frac{1}{n}\boldsymbol{1}_{n\times n}\boldsymbol{K}_t - \frac{1}{n}\boldsymbol{K}\boldsymbol{1}_{n\times n_t} + \frac{1}{n^2}\boldsymbol{1}_{n\times n}\boldsymbol{K}\boldsymbol{1}_{n\times n_t}, \quad (40)$$

where $\boldsymbol{1} := [1, 1, \dots, 1]^\top$, $\boldsymbol{K}_t \in \mathbb{R}^{n\times n_t}$ is the not necessarily centered out-of-sample kernel, and $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ is the training kernel.

5.1.3. Out-of-sample Embedding

One can use Eq. (39) to embed the i-th out-of-sample data point $\boldsymbol{x}_i^{(t)}$. For this purpose, $\boldsymbol{x}_i^{(t)}$ should be used in place of $\boldsymbol{x}$ in Eq. (39).

Note that Eq. (39) requires Eq. (40). In MDS and Isomap, $\boldsymbol{K}$ is obtained by the linear kernel, $\boldsymbol{X}^\top\boldsymbol{X}$, and Eq. (31), respectively. Also, the out-of-sample kernel $\boldsymbol{K}_t$ in MDS is obtained by the linear kernel between the training and out-of-sample data, i.e., $\boldsymbol{X}^\top\boldsymbol{X}_t$. In Isomap, the kernel $\boldsymbol{K}_t$ is obtained by centering the geodesic distance matrix (see Eq. (31)) where the geodesic distance matrix between the training and out-of-sample data is used. In calculation of this geodesic distance matrix, merely the training data points, and not the test points, should be used as the intermediate points in paths (Bengio et al., 2004b).

It is shown in (Bengio et al., 2004b, Corollary 1) that using the geodesic distance with only training data as intermediate points, for teh sake of out-of-sample embedding

in Isomap, is equivalent to the *landmark Isomap* method (De Silva & Tenenbaum, 2003):

$$y_k(\boldsymbol{x}) = \frac{1}{2\sqrt{\delta_k}} \sum_{i=1}^{n} v_{ki}(\boldsymbol{D}_{\text{avg}}^{(g)} - \boldsymbol{D}_t^{(g)}(\boldsymbol{x}_i, \boldsymbol{x})), \tag{41}$$

where $\boldsymbol{D}_{\text{avg}}^{(g)}$ denotes the average geodesic distance between the training points and $\boldsymbol{D}_t^{(g)}$ is the geodesic distance between the i-th training point $\boldsymbol{x}_i$ and the out-of-sample point $\boldsymbol{x}$, in which the training set is used for intermediate points. Hence, one can use Eq. (41) for out-of-sample embedding in Isomap.

It is noteworthy that in addition to the out-of-sample extension using eigenfunctions (Bengio et al., 2004b), there exist some other methods for out-of-sample extension of MDS and Isomap (Bunte et al., 2012; Strange & Zwiggelaar, 2011), which we pass by in this paper.

5.2. Out of Sample for Isomap, Kernel Isomap, and MDS Using Kernel Mapping

There is a kernel mapping method (Gisbrecht et al., 2012; 2015) to embed the out-of-sample data in Isomap, kernel Isomap, and MDS. We introduce this method here.

We define a map which maps any data point as $\boldsymbol{x} \mapsto \boldsymbol{y}(\boldsymbol{x})$, where:

$$\mathbb{R}^p \ni \boldsymbol{y}(\boldsymbol{x}) := \sum_{j=1}^{n} \boldsymbol{\alpha}_j \frac{k(\boldsymbol{x}, \boldsymbol{x}_j)}{\sum_{\ell=1}^{n} k(\boldsymbol{x}, \boldsymbol{x}_\ell)}, \tag{42}$$

and $\boldsymbol{\alpha}_j \in \mathbb{R}^p$, and $\boldsymbol{x}_j$ and $\boldsymbol{x}_\ell$ denote the j-th and ℓ-th training data point. The $k(\boldsymbol{x}, \boldsymbol{x}_j)$ is a kernel such as the Gaussian kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}_j) = \exp(\frac{-||\boldsymbol{x} - \boldsymbol{x}_j||_2^2}{2\,\sigma_j^2}), \tag{43}$$

where σ_j is calculated as (Gisbrecht et al., 2015):

$$\sigma_j := \gamma \times \min_i(||\boldsymbol{x}_j - \boldsymbol{x}_i||_2), \tag{44}$$

where γ is a small positive number.

Assume we have already embedded the training data points using MDS (see Section 2), Isomap (see Section 4), or kernel Isomap (see Section 4.2); therefore, the set $\{\boldsymbol{y}_i\}_{i=1}^n$ is available. If we map the training data points, we want to minimize the following least-squares cost function in order to get $\boldsymbol{y}(\boldsymbol{x}_i)$ close to $\boldsymbol{y}_i$ for the i-th training point:

$$\underset{\boldsymbol{\alpha}_j\text{'s}}{\text{minimize}} \quad \sum_{i=1}^{n} ||\boldsymbol{y}_i - \boldsymbol{y}(\boldsymbol{x}_i)||_2^2, \tag{45}$$

where the summation is over the training data points. We can write this cost function in matrix form as below:

$$\underset{\boldsymbol{A}}{\text{minimize}} \quad ||\boldsymbol{Y} - \boldsymbol{K}''\boldsymbol{A}||_F^2, \tag{46}$$

where $\mathbb{R}^{n\times p} \ni \boldsymbol{Y} := [\boldsymbol{y}_1, \dots, \boldsymbol{y}_n]^\top$ and $\mathbb{R}^{n\times p} \ni \boldsymbol{A} := [\boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_n]^\top$. The $\boldsymbol{K}'' \in \mathbb{R}^{n\times n}$ is the kernel matrix whose (i,j)-th element is defined to be:

$$\boldsymbol{K}''(i,j) := \frac{k(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\sum_{\ell=1}^{n} k(\boldsymbol{x}_i, \boldsymbol{x}_\ell)}. \tag{47}$$

The Eq. (46) is always non-negative; thus, its smallest value is zero. Therefore, the solution to this equation is:

$$\begin{aligned} \boldsymbol{Y} - \boldsymbol{K}''\boldsymbol{A} = \boldsymbol{0} &\implies \boldsymbol{Y} = \boldsymbol{K}''\boldsymbol{A} \\ &\overset{(a)}{\implies} \boldsymbol{A} = \boldsymbol{K}''^{\dagger}\,\boldsymbol{Y}, \end{aligned} \tag{48}$$

where $\boldsymbol{K}''^{\dagger}$ is the pseudo-inverse of $\boldsymbol{K}''$:

$$\boldsymbol{K}''^{\dagger} = (\boldsymbol{K}''^{\top}\boldsymbol{K}'')^{-1}\boldsymbol{K}''^{\top}, \tag{49}$$

and (a) is because $\boldsymbol{K}''^{\dagger}\,\boldsymbol{K}'' = \boldsymbol{I}$.

Finally, the mapping of Eq. (42) for the n_t out-of-sample data points is:

$$\boldsymbol{Y}_t = \boldsymbol{K}_t''\,\boldsymbol{A}, \tag{50}$$

where the (i,j)-th element of the out-of-sample kernel matrix $\boldsymbol{K}_t'' \in \mathbb{R}^{n_t\times n}$ is:

$$\boldsymbol{K}_t''(i,j) := \frac{k(\boldsymbol{x}_i^{(t)}, \boldsymbol{x}_j)}{\sum_{\ell=1}^{n} k(\boldsymbol{x}_i^{(t)}, \boldsymbol{x}_\ell)}, \tag{51}$$

where $\boldsymbol{x}_i^{(t)}$ is the i-th out-of-sample data point, and $\boldsymbol{x}_j$ and $\boldsymbol{x}_\ell$ are the j-th and ℓ-th training data points.

6. Landmark MDS and Landmark Isomap for Big Data Embedding

Nystrom approximation, introduced below, can be used to make the spectral methods such as MDS and Isomap scalable and suitable for big data embedding.

6.1. Nystrom Approximation

Nystrom approximation is a technique used to approximate a positive semi-definite matrix using merely a subset of its columns (or rows) (Williams & Seeger, 2001). Consider a positive semi-definite matrix $\mathbb{R}^{n\times n} \ni \boldsymbol{K} \succeq 0$ whose parts are:

$$\mathbb{R}^{n\times n} \ni \boldsymbol{K} = \left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{B} \\ \hline \boldsymbol{B}^\top & \boldsymbol{C} \end{array}\right], \tag{52}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times m}$, $\boldsymbol{B} \in \mathbb{R}^{m\times(n-m)}$, and $\boldsymbol{C} \in \mathbb{R}^{(n-m)\times(n-m)}$ in which $m \ll n$.

The Nystrom approximation says if we have the small parts of this matrix, i.e. $\boldsymbol{A}$ and $\boldsymbol{B}$, we can approximate $\boldsymbol{C}$ and thus the whole matrix $\boldsymbol{K}$. The intuition is as follows. Assume $m = 2$ (containing two points, a and b) and $n = 5$

(containing three other points, c, d, and e). If we know the similarity (or distance) of points a and b from one another, resulting in matrix $\boldsymbol{A}$, as well as the similarity (or distance) of points c, d, and e from a and b, resulting in matrix $\boldsymbol{B}$, we cannot have much freedom on the location of c, d, and e, which is the matrix $\boldsymbol{C}$. This is because of the positive semi-definiteness of the matrix $\boldsymbol{K}$. The points selected in submatrix $\boldsymbol{A}$ are named *landmarks*. Note that the landmarks can be selected randomly from the columns/rows of matrix $\boldsymbol{K}$ and, without loss of generality, they can be put together to form a submatrix at the top-left corner of matrix.

As the matrix $\boldsymbol{K}$ is positive semi-definite, by definition, it can be written as $\boldsymbol{K} = \boldsymbol{O}^\top \boldsymbol{O}$. If we take $\boldsymbol{O} = [\boldsymbol{R}, \boldsymbol{S}]$ where $\boldsymbol{R}$ are the selected columns (landmarks) of $\boldsymbol{O}$ and $\boldsymbol{S}$ are the other columns of $\boldsymbol{O}$. We have:

$$\boldsymbol{K} = \boldsymbol{O}^\top \boldsymbol{O} = \begin{bmatrix} \boldsymbol{R}^\top \\ \boldsymbol{S}^\top \end{bmatrix} [\boldsymbol{R}, \boldsymbol{S}] \tag{53}$$

$$= \begin{bmatrix} \boldsymbol{R}^\top \boldsymbol{R} & \boldsymbol{R}^\top \boldsymbol{S} \\ \boldsymbol{S}^\top \boldsymbol{R} & \boldsymbol{S}^\top \boldsymbol{S} \end{bmatrix} \overset{(52)}{=} \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^\top & \boldsymbol{C} \end{bmatrix}. \tag{54}$$

Hence, we have $\boldsymbol{A} = \boldsymbol{R}^\top \boldsymbol{R}$. The eigenvalue decomposition (Ghojogh et al., 2019a) of $\boldsymbol{A}$ gives:

$$\boldsymbol{A} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{U}^\top \tag{55}$$

$$\implies \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{U}^\top \implies \boldsymbol{R} = \boldsymbol{\Sigma}^{(1/2)} \boldsymbol{U}^\top. \tag{56}$$

Moreover, we have $\boldsymbol{B} = \boldsymbol{R}^\top \boldsymbol{S}$ so we have:

$$\boldsymbol{B} = (\boldsymbol{\Sigma}^{(1/2)} \boldsymbol{U}^\top)^\top \boldsymbol{S} = \boldsymbol{U} \boldsymbol{\Sigma}^{(1/2)} \boldsymbol{S}$$

$$\overset{(a)}{\implies} \boldsymbol{U}^\top \boldsymbol{B} = \boldsymbol{\Sigma}^{(1/2)} \boldsymbol{S} \implies \boldsymbol{S} = \boldsymbol{\Sigma}^{(-1/2)} \boldsymbol{U}^\top \boldsymbol{B}, \tag{57}$$

where (a) is because $\boldsymbol{U}$ is orthogonal (in the eigenvalue decomposition). Finally, we have:

$$\boldsymbol{C} = \boldsymbol{S}^\top \boldsymbol{S} = \boldsymbol{B}^\top \boldsymbol{U} \boldsymbol{\Sigma}^{(-1/2)} \boldsymbol{\Sigma}^{(-1/2)} \boldsymbol{U}^\top \boldsymbol{B}$$

$$= \boldsymbol{B}^\top \boldsymbol{U} \boldsymbol{\Sigma}^{-1} \boldsymbol{U}^\top \boldsymbol{B} \overset{(55)}{=} \boldsymbol{B}^\top \boldsymbol{A}^{-1} \boldsymbol{B}. \tag{58}$$

Therefore, Eq. (52) becomes:

$$\boldsymbol{K} \approx \left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{B} \\ \hline \boldsymbol{B}^\top & \boldsymbol{B}^\top \boldsymbol{A}^{-1} \boldsymbol{B} \end{array}\right]. \tag{59}$$

Proposition 5. *By increasing m, the approximation of Eq. (59) becomes more accurate. If rank of $\boldsymbol{K}$ is at most m, this approximation is exact.*

Proof. In Eq. (58), we have the inverse of $\boldsymbol{A}$. In order to have this inverse, the matrix $\boldsymbol{A}$ must not be singular. For having a full-rank $\boldsymbol{A} \in \mathbb{R}^{m \times m}$, the rank of $\boldsymbol{A}$ should be m. This results in m to be an upper bound on the rank of $\boldsymbol{K}$ and a lower bound on the number of landmarks. In practice, it is recommended to use more number of landmarks for more accurate approximation but there is a trade-off with the speed. □

Corollary 2. *As we usually have $m \ll n$, the Nystrom approximation works well especially for the low-rank matrices (Kishore Kumar & Schneider, 2017). Usually, because of the manifold hypothesis, data fall on a submanifold; hence, usually, the kernel (similarity) matrix or the distance matrix has a low rank. Therefore, the Nystrom approximation works well for many kernel-based or distance-based manifold learning methods.*

6.2. Using Kernel Approximation in Landmark MDS

Consider Eq. (52) or (59) as the partitions of the kernel matrix $\boldsymbol{K}$. Note that the (Mercer) kernel matrix is positive semi-definite so the Nystrom approximation can be applied for kernels.

Recall that Eq. (14) decomposes the kernel matrix into eigenvectors and then Eq. (10) embeds data. However, for big data, the eigenvalue decomposition of kernel matrix is intractable. Therefore, using Eq. (55), we decompose an $m \times m$ submatrix of kernel. Comparing Eqs. (15) and (53) shows that:

$$\mathbb{R}^{n \times n} \ni \boldsymbol{Y} = [\boldsymbol{R}, \boldsymbol{S}] \overset{(a)}{=} [\boldsymbol{\Sigma}^{(1/2)} \boldsymbol{U}^\top, \boldsymbol{\Sigma}^{(-1/2)} \boldsymbol{U}^\top \boldsymbol{B}], \tag{60}$$

where (a) is because of Eqs. (56) and (57) and the terms $\boldsymbol{U}$ and $\boldsymbol{\Sigma}$ are obtained from Eq. (55). The Eq. (60) gives the approximately embedded data, with a good approximation. This is the embedding in *landmark MDS* (De Silva & Tenenbaum, 2003; 2004). Truncating this matrix to have $\boldsymbol{Y} \in \mathbb{R}^{p \times n}$, with top p rows, gives the p-dimensional embedding of the n points.

Comparing Eq. (60) with Eq. (10) shows that the formulae for embedding of landmarks, $\boldsymbol{R}$, and the whole data (without Nystrom approximation) are similar to each other but one is with only landmarks and the other is with the whole data.

6.3. Using Distance Matrix in Landmark MDS

If $\boldsymbol{D}_{ij}$ denotes the (i, j)-th element of the distance matrix and $\boldsymbol{v}_j$ is the j-th element of a vector $\boldsymbol{v}$, Eq. (13) can be restated as (Platt, 2005):

$$\boldsymbol{K} = \frac{-1}{2} \Big(\boldsymbol{D}_{ij}^2 - \boldsymbol{1}_j \sum_i \boldsymbol{c}_i \boldsymbol{D}_{ij}^2 - \boldsymbol{1}_i \sum_j \boldsymbol{c}_j \boldsymbol{D}_{ij}^2 + \sum_{i,j} \boldsymbol{c}_i \boldsymbol{c}_j \boldsymbol{D}_{ij}^2 \Big), \tag{61}$$

where $\sum_i \boldsymbol{c}_i = 1$.

Let the partitions of the distance matrix be:

$$\mathbb{R}^{n \times n} \ni \boldsymbol{D} = \left[\begin{array}{c|c} \boldsymbol{E} & \boldsymbol{F} \\ \hline \boldsymbol{F}^\top & \boldsymbol{G} \end{array}\right], \tag{62}$$

where $\boldsymbol{E} \in \mathbb{R}^{m \times m}$, $\boldsymbol{F} \in \mathbb{R}^{m \times (n-m)}$, and $\boldsymbol{G} \in \mathbb{R}^{(n-m) \times (n-m)}$ in which $m \ll n$. Comparing Eqs. (52)

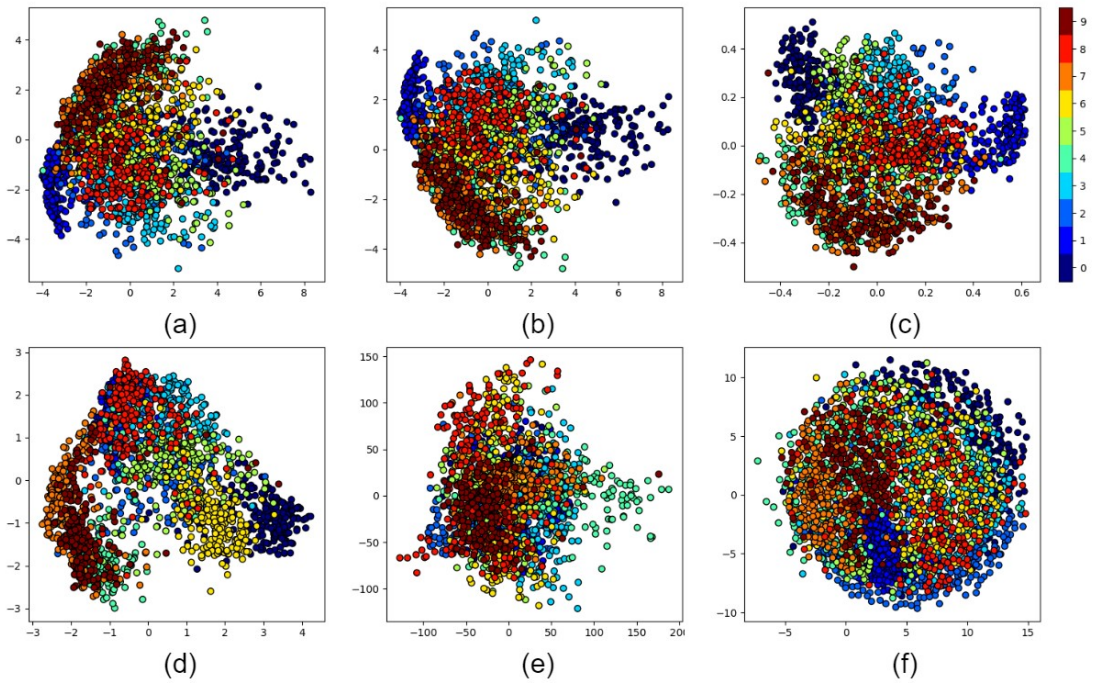

Figure 2. Embedding of the training data in (a) classical MDS, (b) PCA, (c) kernel classical MDS (with cosine kernel), (d) Isomap, (e) kernel Isomap, and (f) Sammon mapping.

and (62) shows that the partitions of the kernel matrix can be obtained from the partitions of the distance matrix as (Platt, 2005):

$$\boldsymbol{A}_{ij} = \frac{-1}{2}\Big(\boldsymbol{E}_{ij}^2 - \mathbf{1}_i \frac{1}{m}\sum_p \boldsymbol{E}_{pj}^2 - \mathbf{1}_j \frac{1}{m}\sum_q \boldsymbol{E}_{iq}^2 + \frac{1}{m^2}\sum_{p,q} \boldsymbol{E}_{pq}^2\Big), \tag{63}$$

$$\boldsymbol{B}_{ij} = \frac{-1}{2}\Big(\boldsymbol{F}_{ij}^2 - \mathbf{1}_i \frac{1}{m}\sum_p \boldsymbol{F}_{qj}^2 - \mathbf{1}_j \frac{1}{m}\sum_q \boldsymbol{E}_{iq}^2\Big), \tag{64}$$

and $\boldsymbol{C}_{ij}$ can be obtained from Eq. (58).

In landmark MDS and landmark Isomap, the partitions (submatrices) $\boldsymbol{E}$ and $\boldsymbol{F}$ of the Euclidean and geodesic distance matrices are calculated, respectively (see Eq. (62)). Then, Eqs. (63), (64), and (58) give us the partitions of the kernel matrix. Eqs. (55) and (60) provide the embedded data.

It is noteworthy that the paper (Platt, 2005) shows that different landmark MDS methods, such as *Landmark MDS (LMDS)* (De Silva & Tenenbaum, 2003; 2004), *FastMap* (Faloutsos & Lin, 1995), and *MetricMap* (Wang et al., 1999) are reduced to landmark MDS introduced here. The landmark MDS is also referred to as the *sparse MDS* (De Silva & Tenenbaum, 2004). Moreover, the *Landmark Isomap (L-Isomap)* (De Silva & Tenenbaum, 2003) is reduced to the landmark Isomap method explained here (see (Bengio et al., 2004b, Corollary 1) for proof). In other words, the large-scale manifold learning methods make use of the Nystrom approximation (Talwalkar et al., 2008).

7. Simulations

7.1. Dataset

For simulations, we used the MNIST dataset (LeCun et al.) includes 60,000 training images and 10,000 test images of size 28×28 pixels. It includes 10 classes for the 10 digits, 0 to 9. Because of tractability of the eigenvalue problem, we used a subset of 2000 training points (200 per class) and 500 test points (50 per class).

7.2. Training Embedding

7.2.1. Classical MDS and Comparison to PCA

The embedding of training data by classical MDS is shown in Fig. 2. As can be seen, this embedding is interpretable because, for example, the digits (7 and 9), (6 and 8), and (5 and 6), which can be converted to each other by slight changes, are embedded close to one another.

Figure 2 also depicts the embedding of training data by PCA. As can be seen, the embedding of PCA is equivalent to the embedding of classical MDS because rotation and flipping does not matter in manifold learning. This vali-

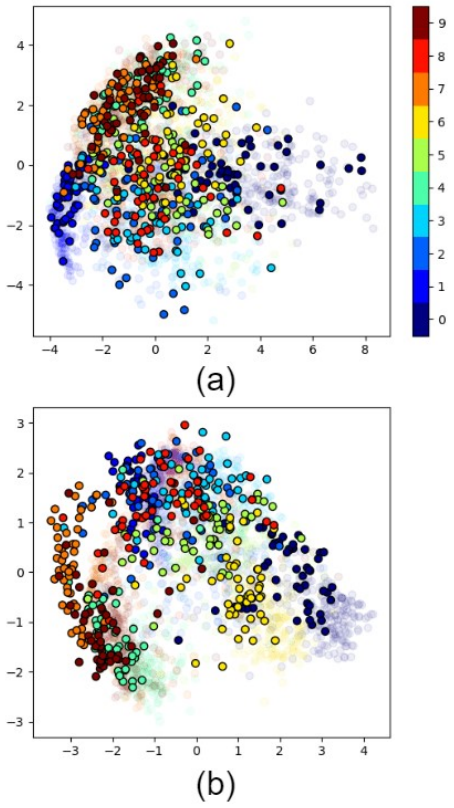

Figure 3. Embedding of the out-of-sample data in (a) classical MDS, and (b) Isomap. The transparent points indicate the embedding of training data.

dates the claim of equivalence of PCA and classical MDS, stated in Section 2.1.3.

7.2.2. Kernel classical MDS, Isomap, Kernel Isomap, and Sammon Mapping

The embedding of kernel classical MDS or the generalized classical MDS (with cosine kernel) is also shown in Fig. 2. This figure also includes the embedding by Isomap. It is empirically observed that the embeddings by Isomap are usually like the legs of an octopus (Ghojogh et al., 2019c). In this embedding, you can see two legs one of which is bigger than the other. Figure 2 also shows the embedding by kernel Isomap. Note that kernel Isomap still uses the kernel calculated using the geodesic distance. Finally, the embedding by Sammon mapping, with 1000 iterations, is also illustrated in Fig. 2. The embeddings by all these methods are meaningful because the more similar digits have been embedded close to each other.

An important fact about the embeddings is that the mean is zero in the embeddigns by classical MDS, kernel classical MDS, Isomap, and kernel Isomap. This is because of double centering the distance matrices in these methods (see Eqs. (12), (13), (31), and (32)).

7.3. Out-of-sample Embedding

The out-of-sample embedding of the classical MDS and Isomap can be seen in Fig. 3. For the out-of-sample embeddings by classical MDS and Isomap, we used Eqs. (39) and (41), respectively. In the Isomap method, as it is difficult to implement the geodesic distance matrix calculated from only the training points as the intermediate points, we used an approximation in which the test points can also be used as intermediate points. A slight shift in the mean of out-of-sample embedding in the Isomap result is because of this approximation.

7.4. Code Implementations

The Python code implementations of simulations can be found in the repositories of the following github profile: `https://github.com/bghojogh`

8. Conclusion

This tutorial and survey paper was on MDS, Sammon mapping, and Isomap. Classical MDS, kernel classical MDS, metric MDS, and non-metric MDS were explained as categories of MDS. Sammon mapping and Isomap were also explained as special cases of metric MDS and kernel classical MDS. Kernel Isomap was also introduced. Out-of-sample extensions of these methods using eigenfunctions and kernel mapping were also provided. Landmark MDS and landmark Isomap using Nystron approximation were also covered in this paper. Finally, some simulations were provided to show the embeddings.

Some specific methods, based on MDS and Isomap were not covered in this paper for the sake of brevity. some examples of these methods are supervised Isomap (Wu & Chan, 2004), robust kernel Isomap (Choi & Choi, 2007) and kernel Isomap for noisy data (Choi & Choi, 2005).

References

Agarwal, Sameer, Wills, Josh, Cayton, Lawrence, Lanckriet, Gert, Kriegman, David, and Belongie, Serge. Generalized non-metric multidimensional scaling. In *Artificial Intelligence and Statistics*, pp. 11–18, 2007.

Aubin, Thierry. *A course in differential geometry*, volume 27. American Mathematical Society, Graduate Studies in Mathematics, 2001.

Beals, Richard, Krantz, David H, and Tversky, Amos. Foundations of multidimensional scaling. *Psychological review*, 75(2):127, 1968.

Bengio, Yoshua, Vincent, Pascal, Paiement, Jean-François, Delalleau, O, Ouimet, M, and LeRoux, N. Learning eigenfunctions of similarity: linking spectral clustering

and kernel PCA. Technical report, Technical Report 1232, Departement dInformatique et Recherche Oprationnelle , 2003a.

Bengio, Yoshua, Vincent, Pascal, Paiement, Jean-François, Delalleau, Olivier, Ouimet, Marie, and Le Roux, Nicolas. *Spectral clustering and kernel PCA are learning eigenfunctions*, volume 1239. Citeseer, 2003b.

Bengio, Yoshua, Delalleau, Olivier, Roux, Nicolas Le, Paiement, Jean-François, Vincent, Pascal, and Ouimet, Marie. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural computation*, 16(10): 2197–2219, 2004a.

Bengio, Yoshua, Paiement, Jean-françcois, Vincent, Pascal, Delalleau, Olivier, Roux, Nicolas L, and Ouimet, Marie. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in neural information processing systems*, pp. 177–184, 2004b.

Bengio, Yoshua, Delalleau, Olivier, Le Roux, Nicolas, Paiement, Jean-François, Vincent, Pascal, and Ouimet, Marie. Spectral dimensionality reduction. In *Feature Extraction*, pp. 519–550. Springer, 2006.

Borg, Ingwer and Groenen, Patrick JF. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

Bunte, Kerstin, Biehl, Michael, and Hammer, Barbara. A general framework for dimensionality-reducing data visualization mapping. *Neural Computation*, 24(3):771–804, 2012.

Cailliez, Francis. The analytical solution of the additive constant problem. *Psychometrika*, 48(2):305–308, 1983.

Choi, Heeyoul and Choi, Seungjin. Kernel Isomap. *Electronics letters*, 40(25):1612–1613, 2004.

Choi, Heeyoul and Choi, Seungjin. Kernel Isomap on noisy manifold. In *Proceedings. The 4th International Conference on Development and Learning, 2005*, pp. 208–213. IEEE, 2005.

Choi, Heeyoul and Choi, Seungjin. Robust kernel Isomap. *Pattern Recognition*, 40(3):853–862, 2007.

Cormen, Thomas H, Leiserson, Charles E, Rivest, Ronald L, and Stein, Clifford. *Introduction to algorithms*. MIT press, 2009.

Cox, Michael AA and Cox, Trevor F. Multidimensional scaling. In *Handbook of data visualization*, pp. 315–347. Springer, 2008.

De Leeuw, Jan. Multidimensional scaling. Technical report, University of California Los Angeles, 2011.

De Silva, Vin and Tenenbaum, Joshua B. Global versus local methods in nonlinear dimensionality reduction. In *Advances in neural information processing systems*, pp. 721–728, 2003.

De Silva, Vin and Tenenbaum, Joshua B. Sparse multidimensional scaling using landmark points. Technical report, Technical report, Stanford University, 2004.

Faloutsos, Christos and Lin, King-Ip. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pp. 163–174, 1995.

Ghodsi, Ali. Dimensionality reduction a short tutorial. Technical report, Department of Statistics and Actuarial Science, Univ. of Waterloo, Ontario, Canada, 2006.

Ghojogh, Benyamin and Crowley, Mark. Unsupervised and supervised principal component analysis: Tutorial. *arXiv preprint arXiv:1906.03148*, 2019.

Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Eigenvalue and generalized eigenvalue problems: Tutorial. *arXiv preprint arXiv:1903.11240*, 2019a.

Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Roweis discriminant analysis: A generalized subspace learning method. *arXiv preprint arXiv:1910.05437*, 2019b.

Ghojogh, Benyamin, Samad, Maria N, Mashhadi, Sayema Asif, Kapoor, Tania, Ali, Wahab, Karray, Fakhri, and Crowley, Mark. Feature selection and feature extraction in pattern analysis: A literature review. *arXiv preprint arXiv:1905.02845*, 2019c.

Ghojogh, Benyamin, Karray, Fakhri, and Crowley, Mark. Quantile-quantile embedding for distribution transformation, manifold embedding, and image embedding with choice of embedding distribution. *arXiv preprint arXiv:2006.11385*, 2020.

Gisbrecht, Andrej, Lueks, Wouter, Mokbel, Bassam, and Hammer, Barbara. Out-of-sample kernel extensions for nonparametric dimensionality reduction. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, volume 2012, pp. 531–536, 2012.

Gisbrecht, Andrej, Schulz, Alexander, and Hammer, Barbara. Parametric nonlinear dimensionality reduction using kernel t-sne. *Neurocomputing*, 147:71–82, 2015.

Gower, John C. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53(3-4):325–338, 1966.

Ham, Jihun, Lee, Daniel D, Mika, Sebastian, and Schölkopf, Bernhard. A kernel view of the dimensionality reduction of manifolds. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 47, 2004.

Hofmann, Thomas, Schölkopf, Bernhard, and Smola, Alexander J. Kernel methods in machine learning. *The annals of statistics*, pp. 1171–1220, 2008.

Holland, Steven M. Non-metric multidimensional scaling (mds). Technical report, Department of Geology, University of Georgia, 2008.

Jung, Sungkyu. Lecture: Multidimensional scaling, advanced applied multivariate analysis. Lecture notes, Department of Statistics, University of Pittsburgh, 2013.

Katsikitis, Mary. The classification of facial expressions of emotion: A multidimensional-scaling approach. *Perception*, 26(5):613–626, 1997.

Kishore Kumar, N and Schneider, Jan. Literature survey on low rank approximation of matrices. *Linear and Multilinear Algebra*, 65(11):2212–2244, 2017.

Kruskal, J. Non-metric multidimensional scaling. a numerical method. *Psychometrika*, 29(1):1, 1964a.

Kruskal, Joseph B. Multidimensional scaling by optimising goodness-of-fit to non-metric hypotheses. *Psychometrika*, 29(1):115–29, 1964b.

LeCun, Yann, Cortes, Corinna, and Burges, Christopher J.C. MNIST handwritten digits dataset. `http://yann.lecun.com/exdb/mnist/`. Accessed: 2019.

Lee, John A and Verleysen, Michel. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

Lee, John Aldo, Lendasse, Amaury, Verleysen, Michel, et al. Curvilinear distance analysis versus isomap. In *European Symposium on Artificial Neural Networks*, volume 2, pp. 185–192, 2002.

Mardia, Kanti V. Some properties of clasical multidimesional scaling. *Communications in Statistics-Theory and Methods*, 7(13):1233–1241, 1978.

Oldford, Wayne. Lecture: Recasting principal components. Lecture notes for Data Visualization, Department of Statistics and Actuarial Science, University of Waterloo, 2018.

Platt, John. Fastmap, metricmap, and landmark mds are all nystrom algorithms. In *AISTATS*, 2005.

Russell, James A and Bullock, Merry. Multidimensional scaling of emotional facial expressions: similarity from preschoolers to adults. *Journal of personality and social psychology*, 48(5):1290, 1985.

Sammon, John W. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 100(5):401–409, 1969.

Saul, Lawrence K and Roweis, Sam T. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of machine learning research*, 4(Jun): 119–155, 2003.

Schlesinger, ItzchakM and Guttman, Louis. Smallest space analysis of intelligence and achievement tests. *Psychological Bulletin*, 71(2):95, 1969.

Strange, Harry and Zwiggelaar, Reyer. A generalised solution to the out-of-sample extension problem in manifold learning. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pp. 471–476, 2011.

Strange, Harry and Zwiggelaar, Reyer. *Open Problems in Spectral Dimensionality Reduction*. Springer, 2014.

Talwalkar, Ameet, Kumar, Sanjiv, and Rowley, Henry. Large-scale manifold learning. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, 2008.

Tenenbaum, Joshua B, De Silva, Vin, and Langford, John C. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Torgerson, Warren S. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

Torgerson, Warren S. Multidimensional scaling of similarity. *Psychometrika*, 30(4):379–393, 1965.

Wang, Jason Tsong-Li, Wang, Xiong, Lin, King-Ip, Shasha, Dennis, Shapiro, Bruce A, and Zhang, Kaizhong. Evaluating a class of distance-mapping algorithms for data mining and clustering. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 307–311, 1999.

Williams, Christopher KI and Seeger, Matthias. Using the Nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pp. 682–688, 2001.

Wu, Yiming and Chan, Kap Luk. An extended Isomap algorithm for learning multi-class manifold. In *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826)*, volume 6, pp. 3429–3433. IEEE, 2004.

Young, Forrest W. *Multidimensional scaling: History, theory, and applications*. Psychology Press, 2013.

Zhao, Xiaoming and Zhang, Shiqing. Facial expression recognition based on local binary patterns and kernel discriminant Isomap. *Sensors*, 11(10):9573–9588, 2011.