# Classical Scaling Revisited

Computer Science Department, Technion, Israel Institute of Technology, Haifa 32000
Gil Shamai
http://www.cs.technion.ac.il/∼sgils/

Yonathan Aflalo
yaflalo@cs.technion.ac.il

Michael Zibulevsky
http://ie.technion.ac.il/∼mcib/

Ron Kimmel
http://www.cs.technion.ac.il/∼ron/

## Abstract

*Multidimensional-scaling (MDS) is an information analysis tool. It involves the evaluation of distances between data points, which is a quadratic space-time problem. Then, MDS procedures find an embedding of the points in a low dimensional Euclidean (flat) domain, optimizing for the similarity of inter-points distances. We present an efficient solver for Classical Scaling (a specific MDS model) by extending the distances measured from a subset of the points to the rest, while exploiting the smoothness property of the distance functions. The smoothness is measured by the $L_2$ norm of the Laplace-Beltrami operator applied to the unknown distance function. The Laplace Beltrami reflects the local differential relations between points, and can be computed in linear time. Classical-scaling is thereby reformulated into a quasi-linear space-time complexities procedure.*

## 1. Introduction

With recent advances in science and technology the amount of digital information being stored and analyzed constantly expands and is sometimes referred to as *big data*. Along with the growth in size of available data, appears the need for simplification and dimensionality reduction. Methods, such as *principal component analysis* (PCA) [29], *self-organizing map* (SOM) [17], *Local Coordinate Coding* [31] [32], and *multidimensional scaling* (MDS) [5], are data reduction techniques that occupy the minds of researchers, who constantly try to reduce their computational and space complexities. Dimensionality reduction achieved by embedding data into a Euclidean space is often referred to as flattening. For example, in [26] and [12], multidimensional reduction was applied to numerically flatten models of monkeys' cortical surfaces. In [27], [24], and [22], flattening was used for image and video analysis.

A family of distance preserving data flattening techniques is the multidimensional scaling or MDS. These methods attempt to map the data into a low dimensional Euclidean space, while preserving, as much as possible, some affinity measures between each pair of data points. In [14], *classical scaling* was used to map non-rigid curved surfaces into a Euclidean space, such that the geodesic distances between each pair of points is as similar as possible to the Euclidean distance between the corresponding embedded points. It was shown that the embedded set of points, referred as a *canonical form*, is invariant to isometric deformations of the non-rigid object. Canonical forms could thereby be used for non-rigid object matching and classification. Here, we give as an example the construction of such forms that we choose to embed in $\mathbb{R}^3$. We show how to efficiently avoid the need to store the full pairwise distances matrix.

The first step in most distance preserving mapping methods is the computation of all the pairwise distances. When the data lies on a manifold, and geodesic distances need to be computed, this task can be time consuming and in some cases impractical. Efficient procedures such as the fast marching method [16], can compute the distance map between all pairs of points, in time complexity of $O(p^2 \log p)$, where $p$ is the number of data points. The time and space complexities are at least quadratic in the number of points, which prohibits dealing with more than a few thousands of points. Attempts to reduce the space complexity of the distance map were made in the *locally linear embedding* (LLE, [23]) and the *Hessian locally linear embedding* (HLLE, [10]) methods, where only local distances were stored between nearby data points, so that the effective space complexity is $O(p)$. In [3], Belkin and Niyogi suggested to embed data points into the Laplace-Beltrami eigenspace for the purpose of data clustering. There, as well, only the closest neighbors of each data points are considered in order to construct the Laplace-Beltrami operator. That way, the local metric is captured while the full pairwise distance map is not evaluated and the global geometric structured is ignored.

The difficulty to deal with all pairwise distances was realized by De Silva and Tenenbaum in [28]. They suggested to flatten only a subset of landmark data points. These landmarks were then used as anchors for interpolating the rest of the points in the target embedded space. The Nyström method [2] is an efficient technique that can be used to construct low rank approximations of positive semidefinite matrices, using only a few columns chosen randomly from the matrix. In [30], Nyström method was used for approximating the affinity matrix and perform MDS. Instead of random sampling, an incremental sampling scheme was proposed for choosing the columns one by one, such that the variance of the affinity matrix is minimized. [8] used Nyström method for graph drawing, and proposed a different sampling scheme based on the *farthest point strategy*, which we find to be more efficient and provided better approximation results. A regularization term can then be used for the pseudo-inverse computation, which further improves the approximation. The graph shortest path distances are being measured using BFS, which assumes all edges are with equal length. [19] combined Kernel-PCA [25] with Nyström method and showed how it can be efficiently used for mesh segmentation and for finding mesh correspondence.

Spectral MDS (SMDS) [1] translates the classical scaling problem into the spectral domain which allows to significantly reduce the time and space complexities of the flattening procedure. The full distance map is evaluated using interpolation from a small set of sampled points, between which the geodesic distances are computed. Then, the classical scaling problem is solved using matrix decomposition in the spectral domain. Here, inspired by the Spectral-MDS, we develop a novel method for distance interpolation that is more efficient, simpler, and more accurate, and avoids the need to explicitly use the spectral domain. Working in the spacial domain is essentially equivalent to using all of the eigenvectors in the spectral domain and is therefore more accurate. Using the interpolation phase, we show how to reformulate the classical scaling problem so that only small matrices are stored and involved in the computation. The numerical experiments demonstrate the improved accuracy and efficiency compared to [1].

The structure of the paper is as follows. In Section 2 we review the Classical Scaling method. In Section 3 we formulate our distance interpolation technique, and show how to approximate the affinity matrix by decomposing it to smaller matrices. Next, in Section 4 we use the interpolation as part of the classical scaling algorithm and show how to solve it without explicitly computing and storing the full pairwise distance matrix. Finally, in Section 5 we support the proposed method with experimental results, followed by conclusions.

## 2. Review of Classical Scaling

Multidimensional scaling (MDS) methods aim at finding an embedding $Z$ in a low dimensional space $\mathbb{R}^m$ of $p$ points such that the Euclidean distances between their corresponding coordinates $\|z_i - z_j\|_{\mathbb{R}^m}$ are as close as possible to some affinity measure $(D_{ij})$. Classical scaling is one such procedure. It is defined through the minimization problem $\arg_Z \min \left\|ZZ^T + \frac{1}{2}JEJ\right\|_F$, where $E_{ij} = D_{ij}^2$ and $J_{ij} = \delta_{ij} - \frac{1}{p}$, and as usual, $\delta_{ij} = 0$ for $i \neq j$ and $\delta_{ii} = 1$ for all $i$. The solution for this problem is achieved by decomposing the matrix $-\frac{1}{2}JEJ$ into its eigenvalues and eigenvectors matrices $V\Lambda V^T$. Then, by considering the $m$ largest eigenvalues and corresponding eigenvectors in the truncated matrices $\tilde{V}$ and $\tilde{\Lambda}$, the solution is given by $Z = \tilde{V}\tilde{\Lambda}^{\frac{1}{2}}$. The traditional classical scaling algorithm requires the computation of the full $E_{p\times p}$ matrix which is practically impossible to obtain when dealing with more than several thousands of points. When the points lay on a surface, the affinities $D_{ij}$ can be defined as the geodesic distances between the data points. In this case, the values of $D$ are invariant to isometric deformations of the surface, thus revealing its intrinsic geometry. In this paper we focus on this case and define $E_{ij}$ as the squared geodesic distance between points $i$ and $j$. In the next section, we show how to approximate the matrix $E$ from a small subset of its rows, thus significantly reducing the space and time complexities.

## 3. Distances Interpolation

Let $\mathcal{M}$ be a manifold embedded in some $\mathbb{R}^m$ space. Denote by $D(x, y)$ the geodesic distance between $x, y \in \mathcal{M}$. We assume $\mathcal{M}$ is sampled by a set of $p$ points, approximating the smooth surface, and hence $D$ can be approximated by a $p \times p$ matrix. Denote by $E$ the $p \times p$ matrix such that $E_{ij} = D_{ij}^2$. Our goal is to compute $E$, as the first step of the classical MDS.

We next show how to approximate the matrix $E$ from a small subset of its rows. Let $F$ be a $n \times p$ matrix which holds $n$ chosen rows of $E$. The rows selection and the construction of $F$ are discussed in Subsection 3.2. We would like to find a $p \times n$ matrix $M$ such that $E \approx MF$. The task of approximating a matrix by decomposing it into smaller matrices has already been addressed before in several places, see for example [11], [20]. One such method is the *Nyström* method [2], which is simple and accurate way for approximating positive semi-definite matrices. Nevertheless, Nyström like procedures usually do not use any prior knowledge about the matrix to be decomposed. Here, we exploit the fact that the values of $E$ are geodesic distances on the manifold. This additional knowledge allows us have a better estimation of the matrix $M$.

### 3.1. Approximating $E$

We first discuss the continuous case, and then move the discrete one. Let $\{x_i\}_{i=1}^n$ be a set of $n$ landmark points chosen from the manifold, and let $x_0$ be an arbitrary point on the manifold. Denote by $\bar{e} : x \in \mathcal{M} \to \mathbb{R}$ the squared geodesic distance from $x_0$ to any other point $x \in \mathcal{M}$. Denote by $f_i$ the value of $\bar{e}$ at point $x_i$, such that $\bar{e}(x_i) = f_i$, and assume that we know these values. Our goal now is to interpolate the function $\bar{e}(x)$ from its $n$ known values.

We follow the idea of Aflalo et al. presented in [1]. There, $\bar{e}(x)$ was interpolated using the Dirichlet energy minimization

$$\arg_{\bar{e}} \min \mathcal{E}(\bar{e}) \qquad \text{s.t.} \qquad \bar{e}(x_i) = f_i, \tag{1}$$

where

$$\mathcal{E}(\bar{e}) = \int_{x\in\mathcal{M}} \|\nabla \bar{e}(x)\|_2^2 da(x) \tag{2}$$

is the Dirichlet's energy of the function $\bar{e}(x)$. $da(x)$ is the infinitesimal volume element, and the gradient is defined with respect to the manifold. The energy $\mathcal{E}(\bar{e})$ was termed in [1] as *the smoothness measure of* $\bar{e}(x)$. Thus, defining the interpolation as that of finding the smoothest function $\bar{e}(x)$ that satisfies the constraints $\bar{e}(x_i) = f_i$.

Here, we change the smoothness term to be the $L_2$ norm of the Laplace-Beltrami operator applied to the unknown distance function $\bar{e}$. We discuss the advantages of this formulation in Subsection 3.3. The energy now reads

$$\mathcal{E}(\bar{e}) = \int_{x\in\mathcal{M}} (\Delta \bar{e}(x))^2 da(x) = \int_{x\in\mathcal{M}} \left(\bar{l}(x)\right)^2 da(x), \tag{3}$$

where $\Delta \bar{e}(x)$ is the Laplace-Beltrami operator on $\mathcal{M}$ applied to $\bar{e}$, and we denote $\bar{l}(x) = \Delta \bar{e}(x)$.

In the discrete domain, denote by $e$ and $l$ the discretization of $\bar{e}$ and $\bar{l}$. $l$ and $e$ are $p \times 1$ column vectors, related by $l = Le$, where $L_{p\times p}$ is the sparse discrete Laplace-Beltrami operator. Any discretization matrix of the Laplace-Beltrami operator can be used. Here, we choose to use the general form $L = A^{-1}W$, where $A$ is a diagonal matrix such that $A_{ii}$ is the metric infinitesimal volume element, and $W$ is the classical cotangent weights matrix for triangulated surfaces as defined in [21]. For other data-sets, other Laplacian definitions can be used. The energy in its discrete form reads

$$\mathcal{E}(e) = \sum_{i=1}^{p} {l_i}^2 A_{ii} = l^T A l = e^T L^T A L e, \tag{4}$$

and the interpolation problem can be written as

$$e^* = \arg_e \min e^T L^T A L e \qquad \text{s.t.} \qquad Be = f, \tag{5}$$

where $f$ is a $n \times 1$ vector holding the values $\{f_i\}_{i=1}^n$, and $B_{n\times p}$ is a selection matrix, whose rows are a subset of the identity matrix's rows, and hence multiplying $B$ by the vector $e$ extracts the subset $f$ from $e$. An alternative form of the problem using a penalty function instead of constraints is given by

$$e^* = \arg_e \min(e^T L^T A L e + \mu \|Be - f\|^2), \tag{6}$$

where the scalar $\mu$ is sufficiently large. The solution is given by

$$\begin{aligned} M &= (L^T A L + \mu B^T B)^{-1} \mu B^T \\ e^* &= M f. \end{aligned} \tag{7}$$

Recapping, given a vector of known values $f_{n\times 1}$ sampled from $e$, one can compute $M_{p\times n}$ and then $e^* = Mf$, which is a reconstruction of $e$, in the sense of being as smooth as possible while satisfying the above constraints. Figures 1 and 2 demonstrate the reconstruction of $e$ for flat and curved manifolds (where $e$ is treated as a column of the matrix $E$).

Next, let $F$ be a $n \times p$ matrix which holds $n$ chosen rows of $E$. Then, we can simply interpolate all the columns of $E$ simultaneously by $\hat{E} = MF$. As $E$ is symmetric by definition, we symmetrize its reconstruction by

$$\hat{E} = \frac{1}{2}(MF + F^T M^T). \tag{8}$$

Symmetrizing $\hat{E}$ allows us to combine the interpolation with the classical MDS algorithm, as we will see in Section 4. Note that we do not need to store the whole matrix $\hat{E}$, but rather keep only the matrices $M$ and $F$, and thereby reduce the space complexity from $O(p^2)$ to $O(np)$. In the triangulated surfaces we tested, for accurate reconstruction of $E$, it was enough to select $n \approx 50$, where the number of vertices, $p$, approximating the surface was in the range of $10^3$ to $10^6$.

### 3.2. Choosing the set of rows

The matrix decomposition developed in the previous section can be seen as a projection of $E$ on the subspace spanned by the chosen rows. Hence, a good choice of rows would be one that captures the range of $E$ with high accuracy. The *farthest point sampling strategy* is a method for selecting points from a manifold that are far away from each other, and is known to be 2-optimal in sense of covering [15]. The first point is selected at random. Then, at each iteration, the farthest point (in geodesic sense) from the already selected ones is selected.

The geodesic distance computation from a point to the rest of the $p$ surface points can be performed efficiently using, for example, the *fast marching method* [16] for two dimensional triangulated surfaces, or using *Dijkstra's shortest path algorithm* [9] for higher dimensions, both with complexity of $O(p \log p)$. The complexity for choosing $n$ points (and hence also obtaining the geodesic distances from them to the rest of the manifold points) with *farthest point sampling* is thus $O(np \log p)$.

Here, we use the *farthest point sampling strategy* as an efficient way of obtaining $n$ rows of the matrix $E$. We choose $n$ samples from the manifold which corresponds to $n$ rows of $E$, and hold them in the $n \times p$ matrix $F$. Since the chosen samples are far from each other, the corresponding rows are expected to capture most of the information of the matrix. While other existing methods need to store the whole matrix in memory or at least scan it a few times to decide which rows are best to choose, here, we do not need to know the entire matrix in advance. This is a strong advantage which could be exploited in problems related to pairwise geodesic computation.

### 3.3. Formulation justification

As discussed in the previous section, Aflalo et al. have used an energy minimization formulation with a smoothness term to interpolate the matrix $E$. This suggested energy minimization formulation (Equations (1), (2)) has several benefits. (1) It yields a simple matrix decomposition for $E$. (2) The matrix $M$ in the solution $e^* = Mf$ does not depend on the values of $f$ but only on the set of indices of chosen rows. Thus, we can compute $M$ once and interpolate all the columns of $E$ simultaneously. (3) We expect the geodesic distance function $\bar{e}(x)$ to be smooth over the manifold, as nearby points should have similar $\bar{e}(x)$ values. This feature is exploited in the smoothness measure definition.

Aflalo et al. [1] demonstrated state of the art results for the matrix reconstruction and the embedding. Here, we followed their idea, with a few significant modifications. (1) Denote by $F_0$ the $n \times n$ matrix which is the intersection of $F$ and $F^T$ in $E$. In other words, $F_0$ holds all pairwise geodesic distances between the $n$ landmark points. The method in [1] used only the values in $F_0$ for the interpolation task, by first reconstructing $F$ from $F_0$, and then reconstructing $E$ from $F$. The benefit of this formulation is that it results in a symmetric decomposition $\hat{E} = MF_0M^T = M\hat{F}$, while here we need to symmetrize the solution and use an additional trick presented in Section 4 for integration with MDS. Here, we skip the reconstruction of $F$, as the values of $F$ were computed when $F_0$ was computed. (2) In [1] all computations were performed in the spectral domain, while here all computations are done in the spacial domain. In the spectral domain, only part of the eigenvectors of the Laplace-Beltrami is considered (100-300 eigenvectors), while here in the spacial domain, no eigenvectors are omitted. Therefore, the accuracy is clearly better. The accuracy of both methods becomes the same only when using all $p$ eigenvectors of $L$ in the spectral domain. (3) Minimization of the Dirichlet energy results in spikes (see Figure 1). The $L_2$ of the Laplacian is hence a more appropriate smoothness measure, turning to one higher degree of differentiation, while keeping the solution simple.

In Figure 1 we demonstrate the interpolation of an arbitrary column of $E$ from its $n$ values. We use a flat surface, and hence the function is simply the squared Euclidean distance from a point in $\mathbb{R}^2$, $z = x^2 + y^2$, where $x$ and $y$ are the Euclidean coordinates. We compare our suggested Laplacian smoothness measure to the Dirichlet smoothness measure. As can be seen, when using the Dirichlet measure, the function includes sharp discontinuities at the constraint points.

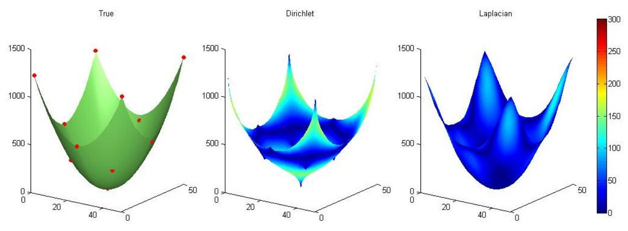


Figure 1: Reconstruction of a column of $E$ on a flat surface. Left: the true values. The chosen $n = 13$ samples are marked with red points. The samples are the constraints $\bar{e}(x_i) = f_i$. Middle and right: the reconstructed columns $e^* = Mf$ using Dirichlet and Laplacian smoothness terms. For comparison, we colored the function according to the absolute error $|e^* - e|$.

In Figure 2, we visualize the reconstruction of a column of $E$ on a curved surface, from its $n$ known values, using our method. As can be seen, the true and reconstructed functions look similar.

## 4. Accelerating Classical Scaling

We are now ready to present an efficient alternative for the classical scaling algorithm using the interpolation discussed in the previous section. Recall that $\hat{E} = \frac{1}{2}(MF + F^TM^T)$ is the approximation of $E$ obtained in Equation (8). A straightforward solution would be similar to classical scaling. Namely, compute $Y = -\frac{1}{2}J\hat{E}J$, decompose it into $V_1\Lambda_1V_1^T$, and then form the truncated decomposition $\tilde{V}_1\tilde{\Lambda}_1\tilde{V}_1^T$, where $\tilde{\Lambda}_1$ and $\tilde{V}_1$ hold the $m$ largest eigenvalues and $m$ corresponding eigenvectors. The solution is then given by

$$Z = \tilde{V}_1\tilde{\Lambda}_1^{\frac{1}{2}}. \tag{9}$$

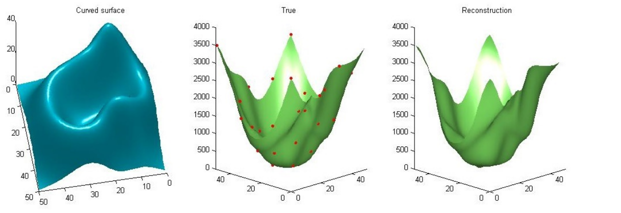


Figure 2: Reconstruction of a column of $e$ on a curved surface. Left: the curved surface. Middle: the true distance function $e$ from the middle point of the surface, sampled in $n = 30$ points marked with red. Right: The interpolation $e^* = Mf$ of the distance function from the samples, using our method.

Since we would like to avoid computing and storing large matrices, the above straightforward procedure would fail to serve this purpose. To that and, we propose an alternative, in which we first decompose $Y$ into smaller matrices and then extract the eigenvectors and eigenvalues from the small matrices.

First, notice that the rank of $Y$ is at most $2n$. This is due to the fact that $F$ is of size $n \times p$, and hence the rank of $MF$ is at most $n$. Consequently, the rank of $\hat{E} = \frac{1}{2}(MF + (MF)^T)$ is at most $2n$. Multiplying by $J$ from both sides, the rank of $Y = -\frac{1}{2}J\hat{E}J$ remains bounded by $2n$. Therefore, $Y$ can be decomposed into matrices that are not larger than $2n \times p$. Such a decomposition could be obtained as follows: Define $S = (M|F^T)$ a horizontal concatenation of the two matrices $M$ and $F^T$. Define the block-permutation matrix $T = \begin{pmatrix} 0_{n\times n} & I_{n\times n} \\ I_{n\times n} & 0_{n\times n} \end{pmatrix}$ where $I$ is the identity matrix. It is easy to verify that

$$\hat{E} = \frac{1}{2}(MF + F^T M^T) = \frac{1}{2}STS^T. \tag{10}$$

Using QR factorization, it is possible to efficiently decompose $JS$ ($p \times 2n$ matrix) into $Q$ ($p \times 2n$ matrix) and $R$ ($2n \times 2n$ matrix), such that $JS = QR$. The columns of $Q$ are orthonormal. $R$ is an upper triangular matrix. We can now write,

$$Y = -\frac{1}{2}J\hat{E}J = -\frac{1}{4}JSTS^TJ = -\frac{1}{4}QRTR^TQ^T. \tag{11}$$

Our next step is to compute an eigenvalue decomposition of $-\frac{1}{4}RTR^T$, that is, $V_2\Lambda_2V_2^T$, and truncate the matrices to get $\tilde{V}_2\tilde{\Lambda}_2\tilde{V}_2^T$, where $\tilde{\Lambda}_2$ and $\tilde{V}_2$ hold the $m$ largest eigenvalues and $m$ corresponding eigenvectors. We have

$$-\frac{1}{4}RTR^T \approx \tilde{V}_2\tilde{\Lambda}_2\tilde{V}_2^T, \tag{12}$$

and hence

$$Y \approx Q\tilde{V}_2\tilde{\Lambda}_2\tilde{V}_2^TQ^T. \tag{13}$$

It is clear that this is the truncated eigenvalue decomposition of $Y$ since $Q\tilde{V}_2$ is orthonormal as a product of orhonormal matrices, and $\tilde{\Lambda}_2$ is diagonal. Therefore, we managed to obtain the truncated eigenvalue decomposition without explicitly computing $Y$. Finally, the solution of the classical scaling problem is given by

$$Z = Q\tilde{V}_2\tilde{\Lambda}_2^{\frac{1}{2}}. \tag{14}$$

We sum up the final MDS acceleration in Procedure 1.

**Procedure 1** *fast-MDS*

```
Input  A manifold $\mathcal{M}$ represented by $p$ vertices, the number
    of samples $n$ and the embedding dimension $m$.
Output  A matrix $Z$ which contains the coordinates of the
    embedding.
1: Compute the Laplace-Beltrami matrix $L = A^{-1}W$.
2: Choose $n$ vertices from $\mathcal{M}$ and compute the matrix $F$,
   using farthest point sampling.
3: Compute $M$ according to Equation (7), where $B$ selects
   the set of chosen vertices.
4: Define $T$, $S$ according to Section 4, and $J$ according to
   Section 2.
5: Compute the QR factorization $JS = QR$.
6: Compute $\tilde{V}_2$ and $\tilde{\Lambda}_2$, which contain the $m$ largest eigen-
   values and corresponding eigenvectors of $-\frac{1}{4}RTR^T$,
   using eigenvalue decomposition.
7: Return the coordinates matrix $Z = Q\tilde{V}_2\tilde{\Lambda}_2^{\frac{1}{2}}$.
```

## 5. Experimental Results

Throughout this section, we compare our proposed method which we term *Fast-MDS* (FMDS) with the following related dimensionality reduction methods: Spectral MDS (SMDS [1]), locally linear embedding (LLE, [23]), Laplacian Eigenmaps [3], the method suggested in [28] termed Landmark-Isomap, the Algorithm proposed by [19], which uses the Nyström method to efficiently perform Kernel-PCA with gaussian radial basis function (termed here as KPCA), the method presented by [8], which is termed SSDE, and finally the method presented by [30], which is termed IS-MDS. All above methods were mentioned in the introduction. We also compare the matrix approximation we developed in Section 3 with the Nyström matrix approximation [2]. Throughout this section we use the Cat, David, Lioness, Centaur and Horse shapes from the TOSCA database [6]. Each shape contains 3400 vertices unless specified otherwise. When using SMDS, we use 200 eigenvectors. The parameter $\mu$ is set to 50.

When using MDS to flatten the intrinsic geometry of a surface into a Euclidean space, the output is known as a canonical form [14]. This form is invariant to isometric deformations of the surface. In our first example, Figure 3, we show the canonical forms of David and Cat shapes, obtained via fast-MDS. This demonstrates the idea that the canonical forms are invariant to isometric deformations of the non-rigid surface.

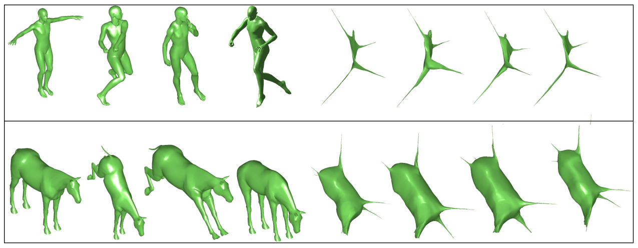


Figure 3: Shapes (left) from the TOSCA database [6] and their corresponding canonical forms (right) obtained by fast-MDS

In our next experiment, Figure 4, we measure the reconstruction error of the squared affinity matrix $E$ of the Cat shape, defined as $\frac{1}{p^2}\|\hat{E} - E\|_F^2$. The Cat shape is chosen for the demonstration of figures 5 and 4 as well. The results are similar for all shapes we have experimented with. Here, we refer by *best M* to the best reconstruction $\hat{E} = MF$ with respect to $M$, given by $\min_M \frac{1}{p^2}\|E - MF\|_F^2$. In addition, we replace the matrix approximation developed in Section 3 with Nyström approximation and refer to this error plot as *Nystrom*.

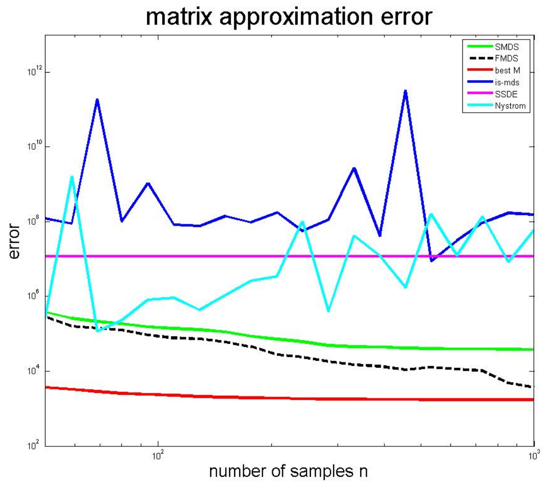


Figure 4: The reconstruction error of the affinity matrix $E$ with respect to the number of samples $n$, using different methods.

In Figure 5 we present the final embedding error obtained by different methods. This error is measured by $stress(Z) - stress(Z^*)$, where

$$stress(Z) = \frac{1}{p^2}\|ZZ^T + \frac{1}{2}JEJ\|_F, \tag{15}$$

$Z$ is the obtained embedding, and $Z^*$ is the embedding obtained by full MDS. In the Landmark-Isomap (LM-ISOMAP) method, a group of landmarks is first selected and embedded using classical MDS. Then, the rest of the points are projected onto the subspace spanned by the embedded landmarks. This method is effective but limited, since the embedding subspace is determined only by the landmark points.

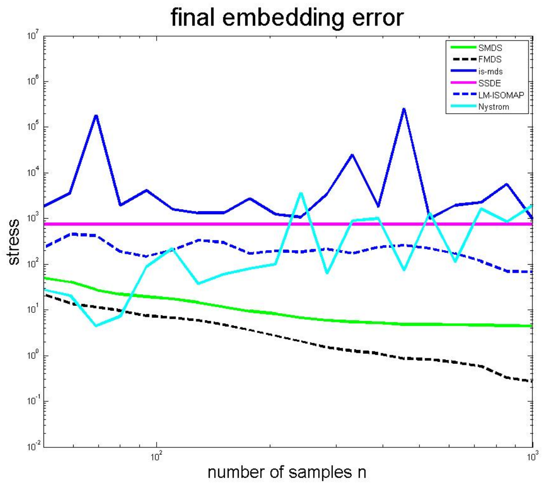


Figure 5: The embedding error of different methods with respect to the number of samples $n$, defined in Equation (15).

Figure 6 compares visually between canonical forms derived from full classical MDS, fast-MDS, spectral MDS, IS-MDS and SSDE. As can be seen, fast-MDS provides a similar canonical form to the full MDS. The corresponding stress error defined in Equation (15) is displayed under each canonical form.

In Figure 7 we visualize the classification of 1200 digits from the MNIST database [18]. Each digit is represented by an $28 \times 28$ image and can be treated as a point in a high dimensional space $\mathbb{R}^{784}$. We connect each point to its K nearest neighbors, so that we get a connected manifold in $\mathbb{R}^{784}$. The distance between neighbor images is calculated using a simple $L_2$ norm, and the geodesic distance between far images is calculated using *Dijkstra's shortest path algorithm* [9]. We apply MDS and the proposed fast-MDS method and flatten the data into a two dimensional space. For visual comparison between the two methods, the classical-MDS embedding points were colored according to their horizontal location, and then the fast-MDS points were colored with

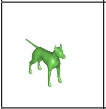
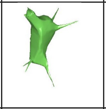
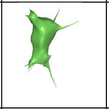
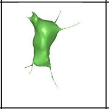
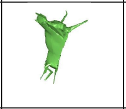
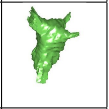
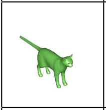
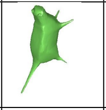
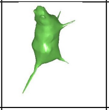
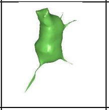
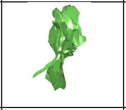
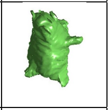
| | MDS | FMDS | SMDS | IS-MDS | SSDE |
|---|---|---|---|---|---|
|  |  |  |  |  |  |
| **Stress** | 344.4 | 366.9 | 389.6 | 651.9 | 892.1 |
|  |  |  |  |  |  |
| **Stress** | 261.1 | 282.8 | 316.4 | 2095.8 | 1016.9 |

Figure 6: Canonical forms of Dog and Cat, using $n = 50$ samples for the compared methods. Left to right: The original shape followed by canonical forms obtained by classical-MDS, fast-MDS, spectral-MDS, IS-MDS and SSDE. The stress error $\frac{1}{p^2}\left\|ZZ^T + \frac{1}{2}JEJ\right\|_F$ of the embedding is displayed at the bottom of the corresponding form.

same colors for same digits. As can be seen, the embedding of FMDS is similar to full MDS, and the proposed method works for high dimensional manifolds.

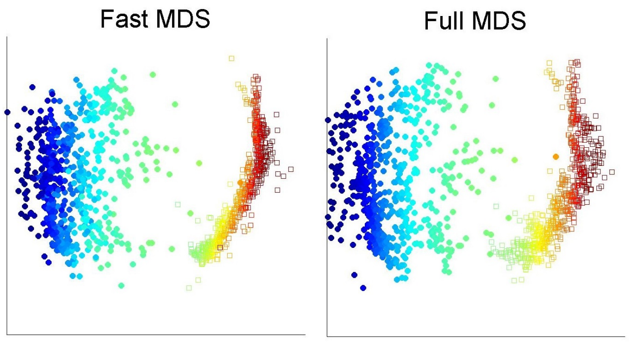


Figure 7: Flat embedding of 1200 images of zeros and ones from the MNIST database, using full MDS and the proposed Fast-MDS. Zeros are marked with filled circles and ones are marked with empty squares.

Multidimensional Scaling can be used to visualize the relations between data points, represented by a distance matrix. In the following example, we use MDS to visualize the relations between the canonical forms of 61 nonrigid shapes. We computed each one of the canonical forms using FMDS, KPCA, LLE and Laplacian Eigenmaps. Then, we aligned each pair of canonical forms using the *iterative closest point algorithm* (ICP [4], [7]), which finds the rigid transformation between two sets of points by minimizing the distance between them. We computed the Euclidean distances between each aligned pair of the 61 canonical forms, and obtained a $61 \times 61$ pairwise Euclidean distance matrix. Finally, we embedded the distance matrix into $\mathbb{R}^2$ using Classical MDS. Figure 8 shows the results. As can be seen, canonical forms computed by MDS, FMDS and SMDS can be used for classification of the nonrigid objects. The other methods mainly consider local relations of the data and hence their canonical forms are less distinctive.

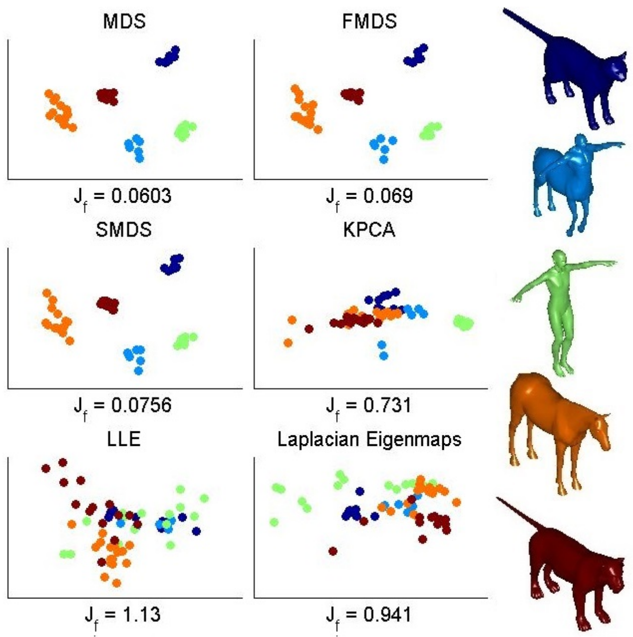


Figure 8: Flat embedding of 61 canonical forms obtained by MDS. The canonical forms are the embedding results of a different isometric poses of Cat, Centaur, David, Horse and Lioness, using MDS, the proposed fast-MDS (FMDS), Spectral-MDS (SMDS), kernel-PCA with nyström (KPCA), locally linear embedding (LLE) and Laplacian Eigenmaps. A quantitative measure of the classification, which is invariant to linear transformation of the data, is computed by $J_f = \text{trace}(S_T^{-1}S_W)$ according to [13]. There, $S_W$ is the within-cluster scatter matrix, $S_B$ is the between-cluster scatter matrix, and $S_T = S_W + S_B$ is the total scatter matrix. The smaller $J_f$ the better is the classification.

Finally, we evaluated the computation time of MDS, FMDS, and SMDS on 5 shapes from the database and then averaged the results. For each shape we created a triangulated mesh with $p$ vertices, and then computed the embedding. The algorithms were evaluated on an i5 Intel computer with 4GB RAM. Figure 9 presents the average time it took each of the methods to compute the result, including the computation of the geodesic distances. The MDS graph was computed only on part of the values due to time and memory limitations. Without memory limitations, its final computation time would have taken more than a couple of hours.

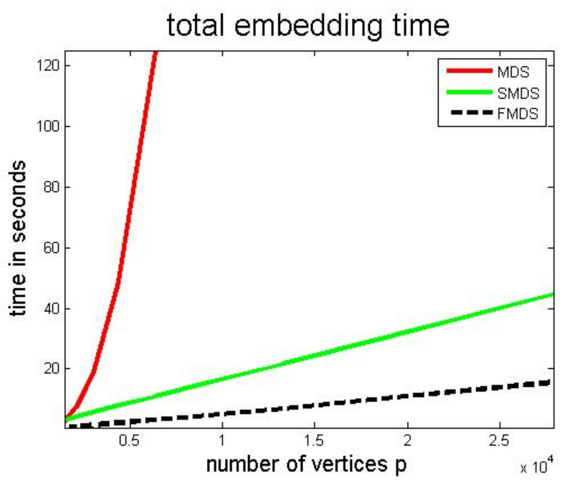


Figure 9: computation time (in seconds) for MDS, fast-MDS, and spectral-MDS on shapes with different number of vertices.

## 6. Conclusions

Using the assumption of a smooth distance function, we were able to reduce the time and space complexities in dimensionality reduction of big data. Following the ideas in *spectral-MDS*, we showed how to split the large distances matrix into two much smaller matrices, which are obtained by solving a simple interpolation problem. Then, we showed how to reformulate the classical scaling problem without explicitly computing the pairwise geodesic distances matrix. The challenging time consuming problem of geodesic distances computation is resolved by sampling the surface in just a few points, and computing the geodesic distances only from these points. As opposed to spectral-MDS, we do not translate the problem into the spectral domain, but rather work in the space domain without truncating of the eigenspace in which we operate. This allows us to save time while improving the accuracy of the original problem. As an example, we demonstrated that an accurate computation of a canonical form, traditionally involving the calculation of all pairwise geodesic distances, which is usually considered a preprocessing stage, can be very efficiently computed on a standard computer.

## 7. Acknowledgements

The research leading to these results has received funding from the European Research Council under European Union's Seventh Framework Programme, ERC Grant agreement no. 267414

## References

[1] Y. Aflalo and R. Kimmel. Spectral multidimensional scaling. *Proceedings of the National Academy of Sciences*, 110(45):18052–18057, 2013. 2, 3, 4, 5

[2] N. Arcolano and P. J. Wolfe. Nyström approximation of wishart matrices. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 3606–3609. IEEE, 2010. 2, 5

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001. 1, 5

[4] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992. 7

[5] I. Borg and P. J. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer, 2005. 1

[6] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. *Numerical geometry of non-rigid shapes*. Springer, 2008. 5, 6

[7] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 2724–2729. IEEE, 1991. 7

[8] A. Civril, M. Magdon-Ismail, and E. Bocek-Rivele. SSDE: Fast graph drawing using sampled spectral distance embedding. In *Graph Drawing*, pages 30–41. Springer, 2007. 2, 5

[9] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik l*, pages 269–27. 4, 6

[10] D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003. 1

[11] P. Drineas, R. Kannan, and M. W. Mahoney. Fast monte carlo algorithms for matrices iii: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1):184–206, 2006. 2

[12] H. A. Drury, D. C. Van Essen, C. H. Anderson, C. W. Lee, T. A. Coogan, and J. W. Lewis. Computerized mappings of the cerebral cortex: a multiresolution flattening method and a surface-based coordinate system. *Journal of cognitive neuroscience*, 8(1):1–28, 1996. 1

[13] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012. 7

[14] A. Elad and R. Kimmel. On bending invariant signatures for surfaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1285–1295, 2003. 1, 5

[15] D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of operations research*, 10(2):180–184, 1985. 3

[16] R. Kimmel and J. A. Sethian. Computing geodesic paths on manifolds. *Proceedings of the National Academy of Sciences*, 95(15):8431–8435, 1998. 1, 4

[17] T. Kohonen. The self-organizing map. *Neurocomputing*, 21(1):1–6, 1998. 1

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6

[19] R. Liu, V. Jain, and H. Zhang. Sub-sampling for efficient spectral mesh processing. In *Advances in Computer Graphics*, pages 172–184. Springer, 2006. 2, 5

[20] M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009. 2

[21] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics*, 2(1):15–36, 1993. 3

[22] R. Pless. Image spaces and video trajectories: Using isomap to explore video sequences. In *ICCV*, volume 3, pages 1433–1440, 2003. 1

[23] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. 1, 5

[24] Y. Rubner and C. Tomasi. *Perceptual metrics for image database navigation*. Springer, 2000. 1

[25] B. Schölkopf, A. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998. 2

[26] E. L. Schwartz, A. Shaw, and E. Wolfson. A numerical solution to the generalized mapmaker's problem: flattening nonconvex polyhedral surfaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11(9):1005–1008, 1989. 1

[27] H. Schweitzer. Template matching approach to content based image indexing by low dimensional euclidean embedding. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 566–571. IEEE, 2001. 1

[28] V. D Silva and J. B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in neural information processing systems*, pages 705–712, 2002. 2, 5

[29] Christpher K.I. Williams. On a connection between kernel PCA and metric multidimensional scaling. In *Advances in Neural Information Processing Systems 13*, pages 675–681. MIT Press, 2001. 1

[30] H. Yu, X. Zhao, X. Zhang, and Y. Yang. ISOMAP using Nyström method with incremental sampling. *Advances in Information Sciences & Service Sciences*, 4(12), 2012. 2, 5

[31] Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *Advances in neural information processing systems*, pages 2223–2231, 2009. 1

[32] Yin Zhou and Kenneth E Barner. Locality constrained dictionary learning for nonlinear dimensionality reduction. *Signal Processing Letters, IEEE*, 20(4):335–338, 2013. 1