Multi-Perspective, Simultaneous Embedding

Md Iqbal Hossain, Vahan Huroyan, Stephen Kobourov, Raymundo Navarrete

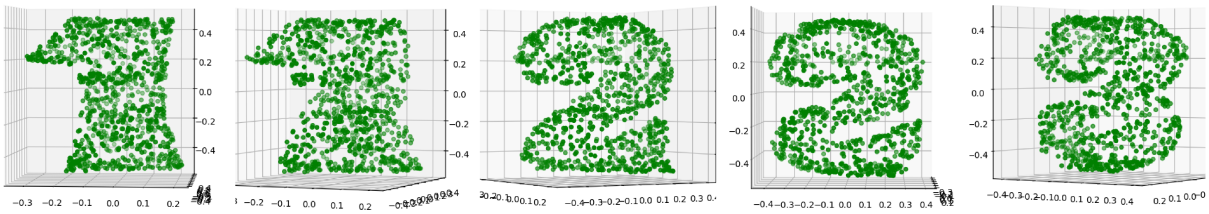

Fig. 1: Given several distances between a set of objects and matching projection planes, MPSE computes 3D coordinates for all the objects, such that when projected to the corresponding 2D planes the distances in each plane match the corresponding input distances. The input dataset is inspired by a sculpture "1, 2, 3" by James Hopkins.

Abstract— We describe MPSE: a Multi-Perspective Simultaneous Embedding method for visualizing high-dimensional data, based on multiple pairwise distances between the data points. Specifically, MPSE computes positions for the points in 3D and provides different views into the data by means of 2D projections (planes) that preserve each of the given distance matrices. We consider two versions of the problem: fixed projections and variable projections. MPSE with fixed projections takes as input a set of pairwise distance matrices defined on the data points, along with the same number of projections and embeds the points in 3D so that the pairwise distances are preserved in the given projections. MPSE with variable projections takes as input a set of pairwise distance matrices and embeds the points in 3D while also computing the appropriate projections that preserve the pairwise distances. The proposed approach can be useful in multiple scenarios: from creating simultaneous embedding of multiple graphs on the same set of vertices, to reconstructing a 3D object from multiple 2D snapshots, to analyzing data from multiple points of view. We provide a functional prototype of MPSE that is based on an adaptive and stochastic generalization of multi-dimensional scaling to multiple distances and multiple variable projections. We provide an extensive quantitative evaluation with datasets of different sizes and using different number of projections, as well as several examples that illustrate the quality of the resulting solutions.

Index Terms—Graph visualization, Dimensionality reduction, Multidimensional scaling, Mental map preservation.

1 Introduction

Given a high dimensional dataset, one of the main visualization goals is to produce an embedding in 2D or 3D Euclidean space, that suitably captures pairwise relationships among the represented data. Dating back to the 1960s, a classical tool that is widely used for both graphs and high dimensional dataset visualization is Multidimensional Scaling (MDS), which aims to preserve the distances between all pairs of datapoints or nodes/objects [45]. Dimensionality reduction is a more general version of this problem, aiming to project a given dataset to a lower dimensional space. There are many popular algorithms for dimensionality reduction, from linear methods such as principal component analysis (PCA) [23], to non-linear methods such as t-SNE [34] which captures local neighborhoods and clusters and UMAP [36] which aims to capture both local and global structure.

We consider situations where instead of just one pairwise distance function, the input is a set of pairwise distance functions on the same set of objects. The optimization goal is to place the points in 3D so that the embedding simultaneously preserves the distances between the objects when projected to some planes. For example, Fig. 2 shows different views of a 3D visualization of a network dataset with multiple attributes. In this visualization, when viewing the 3D coordinates from the correct perspective, the apparent distance between nodes represents the true network distances for one of the attributes. See section 4.4 for details about the dataset and the visualization.

For the case of graph visualisation, consider a set of vertices V (e.g., researchers in one university) and several relationships defined between them E_1, E_2, E_3 (e.g. joint research publications, joint research proposals, membership in different departments). We would like to compute a layout L in 3D as well as 3 planes (P_1, P_2 and P_3) such that when V is projected onto plane P_1 we see the graph $G = (V, E_1)$ so that distances between vertices in the plane P_1 correspond to the distances defined by E_1. Similarly, when L is projected onto plane P_2 we see the graph $G = (V, E_2)$ so that distances between vertices in the plane P_2 correspond to the distances defined by E_2, and the same for P_3 and E_3.

In both settings, this is a strict generalization of the classical MDS problem, which can be seen as a special case with only one pairwise distance function. Even for this special case this problem is known to be difficult as the standard optimization approaches such as gradient descent are not guaranteed to converge to the global optimum. Nevertheless, in practice, when there is a clear structure in the given graph, MDS is often likely to find a good local optimum and as we show in this paper, the simultaneous optimization of our Multi-Perspective, Simultaneous Embedding (MPSE) produces good solutions.

A common approach for visualizing different relationships on the same set of objects involves small multiples and often some mechanism (such as brushing and linking) to connect the same objects in the different views, or morphing from one view to the other. In contrast, MPSE produces one 3D layout and

- *Md Iqbal Hossain and Stephen Kobourov are with the Computer Science Department of The University of Arizona, E-mails: hossain@email.arizona.edu and kobourov@cs.arizona.edu.*
- *Vahan Huroyan and Raymundo Navarrete are with the Department of Mathematics of The University of Arizona, E-mails: vahanhuroyan@math.arizona.edu and raymundo@math.arizona.edu.*

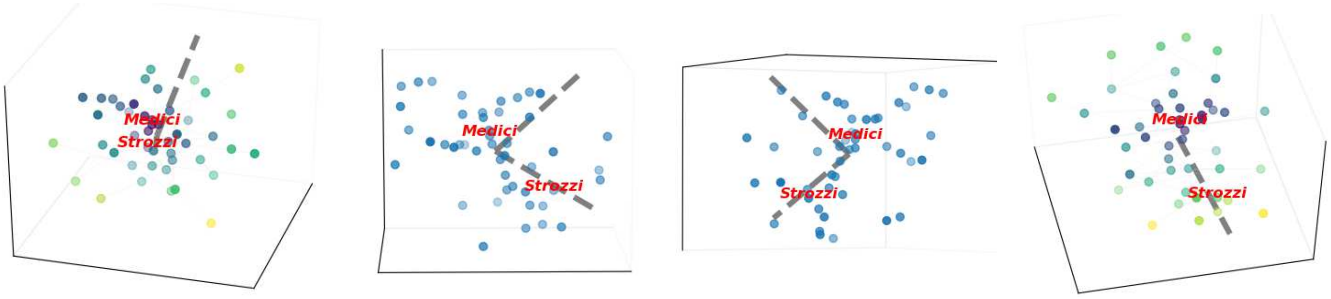

Fig. 2: 3D visualization of marriage and loan relations among prominent families in 15th-century Florence, viewed from different angles. Each node represents one of the families. An edge is drawn between two families if at least one corresponding relation existed between the families. The leftmost image shows the view that best captures pairwise distances of the marriage bonds between the families. The rightmost image shows the view that best captures pairwise distances of the loan bonds between the families. The middle two images show other views of the 3D visualization. The embedding is computed using MPSE with variable projections. A discussion about the quality of the embedding can be found in Section 4.4.

each of the different views is a 2D projection. In this way, MPSE attempts to balance the two main desirable qualities of good visualization of multiple relationships defined on the same set of data: the *readability* of each individual view (typically captured by a faithful embedding in 2D) and *mental map preservation* (typically captured by keeping the objects in the same position across different views). This cannot be accomplished effectively in 2D as there simply is not enough space to realize more than one relationship well, while it becomes plausible in 3D, as our experiments indicate. MPSE can also allow one embedding to tell multiple stories, depending on the different relationships in the data, captured by the corresponding projections, as shown in Figures 2, 9 and 10. Finally, MPSE can be used for reconstructing 3D structure from a collection of 2D images, as shown in Figures 1, 3 and 7.

With the advent of virtual reality and augmented reality systems, 3D visualization and 3D interaction with data in general and graphs in particular are becoming a reality. Still, when presenting 3D results in a paper we are limited to showing 2D snapshots. The MPSE webpage `http://mpse.arl.arizona.edu` provides 3D visualizations with interactive examples, and a functional MPSE prototype.

We describe the MPSE method in detail. We consider two different settings: one where the projection planes are given as part of the input (e.g., the three sides of a 3D cube) and the second where computing the projection planes is part of the optimization. Both settings have been implemented and work well in practice. A fully functional prototype is available on the webpage and source code is available on github. We provide a description on the python implementation and provide several examples of MPSE embeddings with real-world examples. Finally, we provide quantitative evaluation of the scalability of the two variants with increasing number of data points and increasing number of projections.

1.1 Previous work

We review work on visualizing multivariate and multilayer networks, network layout algorithms, multidimensional scaling, simultaneous embedding, and 3D reconstruction algorithms.

Multivariate network visualization. Multivariate [26] and multilayer [19] graph visualization has received a great deal of attention in the last couple of decades. Multi-label, multi-edge, multi-relational, multiplex, multi-modal and many other variants are cleverly encapsulated by the general multilayer network definition of Kivelä et al. [27].

Wattenberg's PivotGraph [50] system can visualize and analyze multivariate graphs not using a global graph layout but rather a grid-based approach focusing on different relationships between node attributes. Semantic substrates [46] unfold multiple attributes of a graph, a pair of attributes at a time, using two dimensions. Pretorius and van Wijk [42] describe an interactive system that relies on clustering of both nodes and edges and interactive exploration using brushing and linking (as well as parallel histograms) to show different graph attributes. *GraphDice* [7] is an interactive multivariate graph visualization system that allows the selection of attributes from an overview plot matrix. This results in a cross dimensional node-link plot for every combination of attributes arranged as a matrix. This system is built on the earlier *ScatterDice* system [15], which provides the ability to interactively explore multidimensional data using scatterplots. Transitions between the scatterplots with one common component are performed as animated rotations in 3D space.

Different from our approach, most of the earlier methods focus on interactive visualizations of multivariate graphs where changing queries result in changing layouts and views. The idea behind our MPSE approach is to produce one 3D layout of the input graph and several projection planes, such that each attribute corresponds to a projection plane in which geometric distances correspond to the graph distances specified by the particular attribute. The main advantage of this approach is that it helps preserve the viewer's 3D mental map, while also capturing different relationships in different projections of the same underlying layout.

Network layout algorithms. Most basic network layouts are obtained using force-directed algorithms. Also known as spring embedders, such algorithms calculate the layout of the underlying graph using only information contained within the structure of the graph itself, rather than relying on domain-specific knowledge [28]. Visual analytics systems for graphs usually focus on interaction [48]. MDS-like approaches to drawing graphs are exemplified in algorithms such as that of Kamada-Kawai [24] and Koren and Carmel [30]. Most commonly used graph drawing systems, such as Graphviz [14], Pajek [6], Tulip [3] and Gephi [5], provide options to visualize graphs in 3D based on MDS-like optimization. Variants of MDS are used in many graph layout systems, including [11, 17, 41, 49]. Other approaches to exploring layouts in 3D include 3D hyperbolic and spherical spaces [12, 29, 37]. None of these earlier approaches however, provides a way to simultaneously optimize different views for the same set of nodes.

Simultaneous embedding. This problem is also related to simultaneous graph embedding and matched drawings of graphs [8]. Specifically, in simultaneous geometric embedding of two or more planar graphs requires planar straight-line drawings of each of the graphs, such that common vertices have the same 2D coordinates in all drawings. This setting is very restrictive [16] and solutions are guaranteed to exist for very

restricted types of input graphs, such as two paths [10], while instances with no solutions can be constructed from a pair of trees [18] or even (path, tree) pairs [2]. Matched drawings require straight-line drawings of the two or more input graphs such that each common vertex has the same y-coordinate in all drawings. Pairs of trees and triples of cycles always have a matched drawing [20]. In general, instances with no solution can be constructed from a pair of planar graphs, or even a (planar graph, tree) pair [13]. Note that matched pairs of drawings can be obtained from the MPSE embedding for every pair of graphs using the intersection line between the corresponding pairs of projection planes as the shared y-coordinate in the pair of matched drawings.

Multidimensional scaling. Consider the problem of recovering the positions (in $\mathbb{R}^p$ for some integer $p > 0$, typically $p = 2$) of a set of objects given their relative pairwise distances. That is, given n objects with indices $1, 2, \ldots, n$, and given pairwise distances $\mathbf{D} = [\mathbf{D}_{ij}]_{i,j=1}^{n}$, where $\mathbf{D}_{ij}$ is the observed distance between objects i and j, we wish to compute positions $x_1, x_2, \ldots, x_n \in \mathbb{R}^p$ such that $\|x_i - x_j\| \approx D_{ij}$ for all objects i and j. The matrix $\mathbf{D}$ may contain distances measured in a higher dimensional Euclidean space, or a more general metric space, may be corrupted by noise, or may just represent a measure of dissimilarity between the objects that does not come from a metric distance. (Metric) multidimensional scaling (MDS) is a well known dimensionality reduction and data visualization technique that can tackle this problem. Its goal is to find an embedding $\mathbf{X} = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^p$ of the n objects so that their pairwise distances agree with the distance or dissimilarity matrix $\mathbf{D}$, by minimizing the stress function

$$\sigma^2(\mathbf{X}; \mathbf{D}) := \sum_{i>j} (\mathbf{D}_{ij} - \|x_i - x_j\|)^2 . \tag{1}$$

Minimizing the stress function (1) is typically accomplished using (stochastic) gradient descent [9] or stress majorization [17].

Note that the original formulation of multidimensional scaling is non-metric MDS. The problem was first studied in the non-metric setting by Shepard [45] and Kruskal [32]. Non-metric MDS recovers structure from measures of similarity, based on the assumption of a reproducible ordering between the distances, rather than relying on the exact distances.

Multi-view multidimensional scaling. Given a *single* pairwise distance or dissimilarity matrix $\mathbf{D}$, MDS aims to find an embedding X that minimizes MDS stress (1). In some applications, data is collected from multiple sources, resulting in *multiple* dissimilarity matrices. The following question arises: Given K dissimilarity matrices $\mathcal{D} = \{\mathbf{D}^1, \mathbf{D}^2, \ldots, \mathbf{D}^K\}$ of the same n objects, how to construct a single embedding X that best represents all of the dissimilarities simultaneously? The answer to this question depends heavily on how the different dissimilarity matrices are related to each other.

In the simplest of cases, the different dissimilarity matrices may correspond to different noisy measurements of the same pairwise relationship. In this case, it may be possible to estimate the true dissimilarity matrix $\mathbf{D}$. For example, if noise in the observed dissimilarity matrices $\mathbf{D}^k$ can be assumed to be independent and identically distributed (i.i.d.) random variables, then the average $(\mathbf{D}^1 + \mathbf{D}^2 + \cdots + \mathbf{D}^K)/K$ is an unbiased estimate of $\mathbf{D}$. It would then be possible to construct an MDS embedding using the estimate of $\mathbf{D}$. Another alternative is to construct an embedding directly, by finding an embedding X that best approximates all of the pairwise dissimilarities simultaneously, for example, by minimizing the functional

$$\sum_{k=1}^{K} \sigma^2(\mathbf{X}; \mathbf{D}^k) = \sum_{k=1}^{K} \sum_{i>j} \left(\mathbf{D}_{ij}^k - \|x_i - x_j\|\right)^2 .$$

Bai et. al [4] propose finding the embedding $\mathbf{X}$ that minimizes the Multi-View Multidimensional Scaling (MVMD) stress function

$$S_{MV}(\mathbf{X}, \alpha; \mathcal{D}) = \sum_{k=1}^{K} \alpha_k^{\gamma} \sum_{i<j} (\mathbf{D}_{ij}^k - \|x_i - x_j\|)^2,$$

where the weights $\alpha = [\alpha_1, \alpha_1, \ldots, \alpha_K]$ are subject to the constraints

$$\sum_{k=1}^{K} \alpha_k = 1, \quad 0 \le \alpha_k \le 1,$$

and $\gamma > 1$ is a fixed parameter. Here the objective is to find an embedding X that minimizes a weighted sum of the MDS stress (1) for the different dissimilarity matrices. The parameter γ balances between just finding the embedding that produces the smallest single stress $\sigma^2(\mathbf{X}; \mathbf{D}^k)$ and assigning equal weights to all of the individual stress values. This idea, along with similar multi-view generalizations of other visualization algorithms, are implemented by Kanaan et al. [25].

In these approaches, the general assumption is that the different dissimilarity matrices correspond to different views of the same relationship in the data. But what if the different dissimilarity matrices truly measure different relationships in the data? In this case, an embedding of the data whose corresponding pairwise distances try to agree with all of the pairwise dissimilarities is not useful. The multi-perspective simultaneous embedding problem that we propose to solve with MPSE is different, as we ask whether it is possible to embed the data so that different perspectives $P^k(X)$ of the embedding X can visualize the different dissimilarities D^k simultaneously.

3D Reconstruction. Our problem is also related to 3D reconstruction from a collection of 2D images. This problem has been widely studied in different settings, including reconstructing the underlying real 3D structure from large collections of 2D photos [47]. More restricted variants are even closer to our setting [31, 43]. Note however, that in our problem we have a constant number of inputs (distance matrices or graphs) and the projections we anticipate can be fixed or computed as a part of the optimization. Our proposed problem and its solution can be viewed also as a step of the structure-from-motion (SfM) problem [38]. The SfM problem is one of the central problems in computer vision which aims to recover the 3D structure of a scene from a set of its projected 2D images. We relate our problem to the SfM problem by assuming some fixed points in the 3D scene which can be estimated on the corresponding 2D projections via some feature extraction algorithm such as SIFT [33]. Once these points are estimated, we can measure the pairwise distance matrices for these points and apply the MPSE algorithm to find the locations of these points in 3D.

1.2 Our Contribution

The main contribution in this paper is the introduction of the multi-perspective simultaneous embedding problem and a generalization of MDS to multiple distance matrices to solve it. This is at the core of the proposed MPSE method for visualizing the same dataset/graph in 3D from several different views, each of which captures a different set of distances/relationships. We consider two main variants: one in which each of the different distances/relationships is associated with a specific 2D projection plane, and the other where computing the projection planes is also part of the optimization. Extensive quantitative experiments show that the method scales well with increasing number of data points and increasing number of projections.

2 Multi-Perspective, Simultaneous Embedding

Our proposed MPSE algorithm is a generalization of the standard (metric) MDS problem. The setting of MDS and its corresponding optimization function, which is called stress function

is discussed in Section 1.1. We remark that if the minimum of the MDS stress function is zero, then the objects can be positioned in $\mathbb{R}^p$ so that their pairwise distances exactly represent the pairwise dissimilarities of $\mathbf{D}$. If the minimum of the MDS stress function is nonzero but not too large, the minimizer of the MDS stress function provides an approximate way to visualize the dissimilarities in $\mathbb{R}^p$.

Suppose now that instead of a single pairwise dissimilarity matrix $\mathbf{D}$, we observe multiple pairwise dissimilarities matrices $\mathcal{D} = \{\mathbf{D}^1, \mathbf{D}^2, \ldots, \mathbf{D}^K\}$ for the same set of n objects. It is natural to ask if MDS can be generalized to produce an embedding so that the dissimilarities $\mathbf{D}^1, \mathbf{D}^2, \ldots, \mathbf{D}^K$ can be visualized by the relative positions of the objects. If it is assumed that the dissimilarities in $\mathcal{D}$ are no more than approximations of some hidden true dissimilarity matrices $\mathbf{D}^1_{true}, \mathbf{D}^2_{true}, \ldots, \mathbf{D}^K_{true}$, then simple generalizations of MDS exist, as described in section 1.1. In all of these generalizations, an embedding $\mathbf{X}$ is produced so that the distances $\|x_i - x_j\|$ are as close as possible to all of the dissimilarity measurements $\mathbf{D}^1_{ij}, \mathbf{D}^2_{ij}, \ldots, \mathbf{D}^K_{ij}$. Nonetheless, this is an undesirable assumption if the goal is to visualize multiple *distinct* relationships between the same objects. In particular, the dissimilarity measurements $\mathbf{D}^1_{ij}, \mathbf{D}^2_{ij}, \ldots, \mathbf{D}^K_{ij}$ can be so different that no embedding X can produce pairwise distances $\|x_i - x_j\|$ that help visualize all of these relations in a meaningful way.

Our generalization of MDS is inspired by the problem of 3D reconstruction from multiple 2D images. We first explain our motivation with an example based on a sculpture of James Hopkins, which illustrates how different the same object can look when observed from a different viewpoints. Depending on the direction from which the viewer sees the sculpture, the viewer will see a different digit '1' or '2' or '3'. For our experiments, we recreated the "1, 2, 3" sculpture and uniformly at random fixed a set of points in the three dimensions that produces the same visual effect when plotted. The set of points forms a figure '1' when viewed from the x-axis, it forms a figure '2' when viewed 60 degrees towards the y-axis, and finally as a figure '3' when viewed another 120 degrees towards the y-axis. Suppose now that the 3D structure is unknown, and the goal is to construct it using only the information from these 3 images. That is, we know the 3 distance matrices $\mathbf{D}^1, \mathbf{D}^2, \mathbf{D}^3$ measuring the distances between the same set of keypoints in each of these images, and the goal is to find an embedding $\mathbf{X}$ so that

$$\mathbf{D}^k_{ij} \approx \|P^k(x_i) - P^k(x_j)\|, \quad k = 1, 2, 3, \quad i, j = 1, 2, \ldots, n,$$

where P^1, P^2, P^3 are the projections that map keypoints in 3D to their corresponding images. For this example one can take the following projection matrices:

$$\mathbf{P}^1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{P}^2 = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{P}^3 = \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

We can ask two questions: 1) given the dissimilarity matrices and the projections, can we recover the embedding $\mathbf{X}$? 2) given the dissimilarity matrices, can we recover the projection matrices and the embedding $\mathbf{X}$? Clearly the latter question is harder to answer. In both cases, we suggest a natural generalization of the MDS stress function

$$\sum_{k=1}^{3} \sum_{i>j} \left(D^k_{ij} - \|P^k(x_i) - P^k(x_j)\| \right)^2.$$

This can be naturally generalized to any number of distance (dissimilarity) matrices and projections. Even if the multiple dissimilarity measures at hand are not related to each other in such a geometric way, a similar idea may still provide a useful visualization of the multiple relations involved.

Motivated by these examples, we generalize the MDS stress function in the following way: Given n objects with L distinct dissimilarity matrices D^l, the multi-perspective MDS stress function is defined as

$$\begin{aligned} S(\mathbf{X}, \mathcal{P}; \mathcal{D}) &= \sum_{k=1}^{L} \sigma^2(P^{(k)}(\mathbf{X}); \mathbf{D}^{(k)}) \\ &= \sum_{k=1}^{L} \sum_{i>j} \left(\mathbf{D}^{(k)}_{ij} - \|P^{(k)}(x_i) - P^{(k)}(x_j)\| \right)^2. \end{aligned} \tag{3}$$

Here, $\mathcal{P} = [P^{(1)}, P^{(2)}, \ldots, P^{(K)}]$, where $P^k : \mathbb{R}^p \to \mathbb{R}^q$ are the mappings (projections). We write $P^{(k)}(X) = [P^{(k)}(x_1), P^{(k)}(x_2, \ldots, P^{(k)}(x_n)]$. We mainly focus on the special case of linear orthogonal mappings, which correspond to orthogonal projections. There are two different ways in which the stress function (3) can be minimized: 1) Given a fixed set of mappings P^l, we find a minimizer $\mathbf{X} \in \mathbb{R}^{n \times p}$; 2) we find a pair $(\mathbf{X}, \mathcal{P})$ minimizing the stress, where $\mathbf{X} \in \mathbb{R}^{p \times n}$ and the mappings P^k for $1 \leq k \leq L$ are restricted to a class of functions (such as linear orthogonal transformations from $\mathbb{R}^p$ to $\mathbb{R}^q$).

2.1 MPSE with Fixed Perspectives

We first consider minimization of the multi-perspective MDS stress function (3) with respect to the embedding $\mathbf{X}$ only:

$$\underset{x_i \in \mathbb{R}^p}{\text{minimize}}\ S(\mathbf{X}, \mathcal{P}; \mathcal{D}) \tag{4}$$

In this setup, the perspective mappings $P^{(1)}, \ldots, P^{(K)} : \mathbb{R}^p \to \mathbb{R}^q$ are predetermined and fixed. We assume that these mappings are differentiable, so that a solution to (4) may be attainable using a gradient descent scheme.

We remark that the objective function for MPSE with fixed projections in (4) is differentiable. Since the multi-perspectives MDS stress function (3) is generally non-convex, minimization to a global minimum is not guaranteed by a vanilla implementation of gradient descent. In addition, computing the full gradient of (3) at each iteration is expensive for large data sets and might make the algorithm infeasible. Thus coming up with a fast and accurate solution is one of the main contributions in this paper. In section 3, we describe a stochastic gradient descent scheme with adaptive learning rate for efficient solution to this problem.

2.2 MPSE with Varying Projections

We again consider minimization of the multi-perspective MDS stress function (3), but we no longer assume predetermined and fixed perspective mappings $\mathcal{P}$. Our goal is then to find both the embedding $\mathbf{X}$ and the perspective mappings $\mathcal{P}$ that best capture the given distance matrices $\mathcal{D}$. In order to do this, a suitable parametric family $\mathcal{O}$ of $C^1(\mathbb{R}^p, \mathbb{R}^q)$ perspective mappings must be defined. The optimization problem becomes

$$\underset{\mathbf{X} \in \mathbb{R}^{p \times n},\ P^{(k)} \in \mathcal{O}}{\text{minimize}}\ S(\mathbf{X}, \mathcal{P}; \mathcal{D}) \tag{5}$$

For ease of exposition, we only consider the set of orthogonal projections for the family of perspective mappings. Treatment of more general families of perspective functions can be found in [21]. By $\mathbb{O}^{3\times 2}$ we denote the family of orthogonal projections from $\mathbb{R}^3$ to $\mathbb{R}^2$. Then, each perspective mapping can be parameterized by an orthogonal $p \times q$ matrix Q. The perspective mapping $P^{(Q)}$ is given by $x \mapsto Qx$. Note that the perspective mappings vary smoothly with respect to changes in the parameters, in the sense that the map $Q \mapsto P^{(Q)}(x)$ is C^1 for every $x \in \mathbb{R}^p$. The optimization problem (5) can then be rewritten as

$$\underset{\mathbf{X} \in \mathbb{R}^{p \times n},\ Q^{(k)} \in \mathbb{O}^{3\times 2}}{\text{minimize}}\ \sum_{k=1}^{n} \sigma^2(Q^{(k)}(\mathbf{X}); \mathcal{D}) \tag{6}$$

We remark that the objective function of MPSE with varying projections in (6) is also differentiable. However, since the set $\mathbb{O}^{3\times 2}$ of 3 by 2 orthogonal matrices is not a subspace of $\mathbb{R}^{3\times 2}$, minimizing (6) with respect to Q_l cannot be accomplished via gradient descent. Instead, we make use of projected gradient descent, where Q_l is updated by first moving towards the direction of steepest descent, and then projecting back onto the set $\mathbb{O}^{3\times 2}$. If $A \in \mathbb{R}^{3\times 2}$ matrix, then the projection of A onto $\mathbb{O}^{3\times 2}$ is the matrix $\Pi(A) \in \mathbb{O}^{3\times 2}$ that minimizes $\|A - Q\|_F$ among all $Q \in \mathbb{O}^{3\times 2}$. There is a simple way to compute $\Pi(A)$: if $U\Sigma V^T$ is the reduced singular value decomposition of A, then $\Pi(A) = UV^T$.

2.3 Some Extensions

The MDS stress function can be extended to general graph structures. That is, we do not require that a pairwise dissimilarity relation exists between every pair of nodes. Moreover, we assume we are given a weighted pairwise relation, where some edges contribute to the stress function more than the others. To make stress values more meaningful, it is common to work with a normalized MDS stress function. The normalized MDS stress function is

$$\tilde{\sigma}^2(\mathbf{X}, \mathbf{D}) = \frac{\sum_{(i,j)\in E} w_{ij} \left(\mathbf{D}_{ij} - \|(x_i) - (x_j)\|\right)^2}{\sum_{(i,j)\in E} w_{ij} (\mathbf{D}_{ij})^2}. \quad (7)$$

where E is the set of edges in the graph for which a dissimilarity relation exits and w_{ij} is the weight corresponding to edge (i, j).

We also generalize MPSE to weighted graphs, where the normalized MPSE stress function becomes

$$\tilde{S}^2(\mathbf{X}, \mathcal{P}; \mathcal{D}) = \frac{1}{K} \sum_{k=1}^{K} \tilde{\sigma}^2(P^{(k)}(\mathbf{X}), \mathbf{D}^{(k)}), \quad (8)$$

where $\mathbf{P}^{(k)}(X) = [\mathbf{P}^{(k)}(x_1), \mathbf{P}^{(k)}(x_2), \ldots, \mathbf{P}^{(k)}(x_N)]$ for each $k = 1, 2, \ldots, K$.

3 Algorithms

In this section we discuss our proposed algorithms for solving the optimization problems for MPSE with fixed projections (4) and MPSE with varying projections (5). In our experiments, we found that convergence to a global minimum of (3) is unlikely using basic gradient descent algorithms. We found that a combination of smart initialization and stochastic gradient descent with adaptive learning rate usually produces better results than the vanilla version of the gradient descent. The benefits of using stochastic gradient are twofold: first, it is faster than the vanilla version of the gradient descent, and second, as observed by others [51], it helps avoid local minima.

Given a dissimilarity matrix $\mathbf{D}$, let $\xi \sim \Xi(\mathbf{D}, c)$ be a random variable returning a dissimilarity matrix ξ, where for each (i, j), $1 \le i, j \le n$ we include $\mathbf{D}_{ij}$ with probability c and otherwise set $\xi_{ij} = 0$. Let $\sigma^2(\mathbf{X}, \xi)$ and $\nabla_{\mathbf{X}}\sigma^2(\mathbf{X}, \xi)$ be unbiased estimates of the MDS stress and its gradient at $\mathbf{X}$. Then, the multi-perspective MDS stress (3) is approximated by

$$S^2_\xi(\mathbf{X}, \mathcal{P}; \mathcal{D}) = S^2(\mathbf{X}, \mathcal{P}, \xi)$$

where $\xi = [\xi^1, \ldots, \xi^K]$ and $\xi^{(k)} \sim \Xi(\mathbf{D}^{(k)}, c)$. The gradients $\nabla_{\mathbf{X}} S^2_\xi(\mathbf{X}, \mathcal{P}; \mathcal{D})$ and $\nabla_{Q^{(k)}} S^2_\xi(\mathbf{X}, \mathcal{P}; \mathcal{D})$ are similarly computed. Note that all these can be computed simultaneously with one pass through the edge list.

We use an adaptive gradient descent framework from [35] with the following adaptive scheme for the learning rate:

$$\mu_{\mathbf{X}}(\mathbf{X}_t, \mathbf{X}_{t-1}, \xi_t) = \frac{(\mathbf{X}_t - \mathbf{X}_{t-1})^T (\nabla_{\mathbf{X}} S^2_{\xi_t}(\mathbf{X}_t - \nabla_{\mathbf{X}} S^2_{\xi_T}(\mathbf{X}_{t-1}))}{\left\|\nabla_{\mathbf{X}} S^2_{\xi_t}(\mathbf{X}_t) - \nabla_{\mathbf{X}} S^2_{\xi_t}(\mathbf{X}_{t-1})\right\|^2} \quad (9)$$

The adaptive learning rate for the projection matrices $\mu_Q(t)$ is computed in a similar fashion.

We summarize the resulting algorithm for MPSE with fixed projections(4) in Algorithm 1. We remark that the problem is non-convex and there is no guarantee for convergence. However, our extensive experiments show that with smart initialization, or with multiple runs of the algorithm (with different random initialization), the algorithm consistently finds good solutions. The idea behind the smart initialization is that even though the problem is non-convex, around the global optimum the problem is well-behaved and so starting close to the global optimum we can apply gradient descent.

Algorithm 1 MPSE with fixed projections (problem (4))

Input: Dissimilarity matrices: $\mathcal{D} = \{\mathbf{D}^1, \ldots, \mathbf{D}^K\}$, initial embedding $\mathbf{X}_0$, initial learning rate μ_0, stochastic constant c, iteration number T.

for $t = 1, 2, \ldots, T$ **do**
  $\xi_t \sim \Xi(\mathbf{D}, c)$
  $\mathbf{X}_t = \mathbf{X}_{t-1} - \mu_{t-1} \nabla_{\mathbf{X}} S^2_{\xi_t}(\mathbf{X}_t, \mathbf{P}; \mathbf{D})$
  $\mu_t = \mu(\mathbf{X}_{t-1}, \mathbf{X}_t, \xi_t)$
end for

Output: $\mathbf{X}_T$

Next, we describe the algorithm for MPSE with variable projections (5) and summarize it in Algorithm 2. We remark that this problem is more complex then MPSE with fixed projections. Variable projections offer more degrees of freedom, at the expense however of non-convex optimization and non-trivial local optima. Extensive experiments in this setting also show that the algorithm works well in practice.

Algorithm 2 MPSE with varying perspectives (problem (5))

Input: Dissimilarity matrices: $\mathcal{D} = \{\mathbf{D}^1, \ldots, \mathbf{D}^K\}$, initial embedding and perspective parameters $\mathbf{X}_0$ and $\mathbf{Q}_0$, initial learning rates $\mu_{\mathbf{X}}$, and μ_Q, stochastic constant c, iteration number T.

for $t = 1, 2, \ldots, T$ **do**
  $\xi_t \sim \Xi(\mathbf{D}, c)$
  Compute $\nabla S^2_\xi(X_{t-1}, \mathbf{Q}_{t-1})$ and $\nabla S^2_\xi(X_t, \mathbf{Q}_t)$
  Compute μ_t.
  $\mathbf{X}_t = \mathbf{X}_{t-1} - \mu_{X,0} \nabla_{\mathbf{X}} S^2_\xi(\mathbf{X}, \mathbf{Q}_t)$.
  $\mathbf{Q}_t = \Pi(\mathbf{Q}_t - \mu_{\mathbf{Q}_t} \nabla_Q S^2_\xi)$
  $\mu^X_t = \mu(\mathbf{X}_{t-1}, \mathbf{X}_t, \xi_t)$
  $\mu^Q_t = \mu(\mathbf{Q}_{t-1}, \mathbf{Q}_t, \xi_t)$
end for

Output: X_{final}, $\mathbf{Q}_{\text{final}}$.

Algorithm 3 summarizes our proposed smart initialization of $\mathbf{X}_0$ and $\mathbf{Q}_0$. We have found that a preliminary computation of the optimal 'combined' MDS embedding of the dissimilarity relations works best. Finally, we remark that the standard practice of multiple runs with different random initialization also helps avoid local minima.

4 Experimental Evaluation

In this section we experimentally evaluate the proposed MPSE algorithms: with fixed projections and with variable projections. Since the problem that we study is new, there are no algorithms to compare against; instead, we create benchmark datasets that we believe can demonstrate and test the MPSE algorithms. For each dataset, we first describe how it was created and show the outputs of MPSE. Note that Algorithm 2, similar to the regular MDS, computes results invariant to global

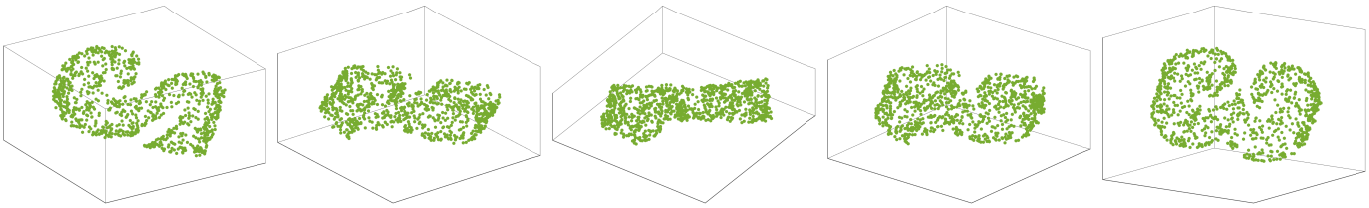

Fig. 3: The output of Algorithm 2 on the distance matrices of the One-Two-Three dataset form Fig. 4. The first subfigure shows the projection which recovers digit 2, the third one corresponds to digit 1, and the last one corresponds to digit 3. The second and fourth shows the rotational transition in 3D between 2 and 1 and between 1 and 3. The final MPSE stress for the One-Two-Three dataset with varying projections (Algorithm 2) is 0.07, with projection-wise stress values of 0.04, 0.08 and 0.08.

Algorithm 3 Initialization for Algorithm (2)

Input: Dissimilarity set $\mathcal{D}$, random initial embedding and perspective parameters X_0 and $\mathcal{Q}_0$, initial learning rates μ_X, and μ_Q, stochastic constant c, iteration number T.
$D_{ij}^2 = \frac{d_1}{d_2 K}\sum_{k=1}^{K}(D_{ij}^{(k)})^2$ ▷ Combine dissimilarities.
for $t = 1, 2, \ldots, T$ **do**
$\xi_t \sim \Xi(D, c)$
$\mathbf{X}_t = \mathbf{X}_{t-1} - \mu_{t-1}\nabla_{\mathbf{X}} S^2_{\xi_t}(\mathbf{X}_t, \mathbf{P}; \mathbf{D})$
$\mu_t = \mu(\mathbf{X}_{t-1}, \mathbf{X}_t, \xi_t)$
end for ▷ Compute MDS embedding of $\mathbf{D}$ on $\mathbb{R}^{d_1}$:
$\mathbf{X} = \mathbf{X}_t$.
for $t = 1, 2, \ldots, T$ **do**
$\xi_t \sim \Xi(\mathbf{D}, c)$
$\mathbf{Q}_t = \mathbf{Q}_{t-1} - \mu_{t-1}\nabla_{\mathbf{Q}} S^2_{\xi_t}(\mathbf{X}_t, \mathbf{Q}_t; \mathcal{D})$
$\mu_t = \mu(\mathbf{X}_{t-1}, \mathbf{X}_t, \xi_t)$
end for ▷ Compute optimal perspective parameters given embedding $\mathbf{X}$:
$\mathbf{X} = \mathbf{X}_t$.
Output: $\mathbf{X}_{\text{final}}$, $\mathbf{Q}_{\text{final}}$.

rotation/reflection and there is no way to recover the exact orientation of the embedding without additional information.

4.1 One-Two-Three Dataset

In the introduction we motivated the MPSE problem using the sculpture "1, 2, 3" by James Hopkins. To generate the dataset for Fig. 1 we attempted to reverse-engineer the sculpture as follows: We created 2D visualizations of the 3 projections by creating images of the digits 1, 2 and 3 and sampled 1000 points from each; see Fig. 4.

Fig. 4: One-Two-Three Dataset: each subfigure contains 1000 points sampled from the original images of digits 1, 2 and 3. The points are labeled from top to bottom to preserve their correct positions.

Then we computed the 2D distances for each set of points and fed those as input to the MPSE algorithm, aiming to recover the 3D figure of the sculpture. The goal is to see whether it is possible to place 1000 points in 3D so that it would look like digits 1, 2 and 3 when seen from different viewpoints.

We first run Algorithm 1 for the distance matrices created from the One-Two-Three dataset; see Fig. 4. For the algorithm we used the following parameters: maximum number of iterations $T = 100$, fixed projections $\mathbf{P}^1, \mathbf{P}^2$ and $\mathbf{P}^3$ from (2), starting learning rate $\mu_0 = 1$ and stochastic constant $c = 0.01$ with random initialization. The results are shown in Fig. 1 and indicate that the algorithm successfully placed the points in 3D such that one can see digits 1, 2 and 3 (see the first, third and fifth subfigures of Fig. 1). The quality of the MPSE embedding in Fig. 1 is also quantified with low stress values: overall stress 0.001, with projection-wise stress values of 0.0009, 0.0008 and 0.0011.

Next, we run Algorithm 2 for the same One-Two-Three dataset, with the following parameters: maximum number of iterations $T = 100$, the starting learning rate $\mu_0 = 1$ and stochastic constant $c = 0.01$ with smart initialization described in Algorithm 3. The results are shown in Fig. 3. We remark that for the MPSE with varying projections the results are correct, up to a global rotation and reflection. The algorithm again successfully placed the points in 3d such that one can see digits 1, 2 and 3 (see the first, third and fifth subfigures of Fig. 3).

4.2 Circle-Square Dataset

The Circle-Square dataset is also an example of 3D share reconstruction from 2D. For this dataset we construct 2D images of a unit square and a circle with radius 1. From these figures we sample 100 points; see Fig. 5. We create the corresponding two distance matrices and run the MPSE algorithms.

Fig. 5: Circle-Square dataset: the input is 100 points sampled from a square and a circle.

We first run Algorithm 1 for the distance matrices created from the Circle-Square dataset; see Fig. 5. For the algorithm we used the following parameters: maximum number of iterations $T = 500$, fixed projections $\mathbf{P}^1$, and $\mathbf{P}^2$ from (2), starting learning rate $\mu_0 = 1$ and stochastic constant $c = 0.01$ with random initialization. The results are shown in the first row of Fig. 7. The algorithm successfully placed the points in 3D so that one can see a circle and a square from different planes (first and last subfigures). The second and third subfigures show the rotational transition in 3D between the circle and the square.

Next, we run Algorithm 2 for the same Circle-Square dataset with the following parameters: maximum number of iterations $T = 100$, the starting learning rate $\mu_0 = 1$ and stochastic

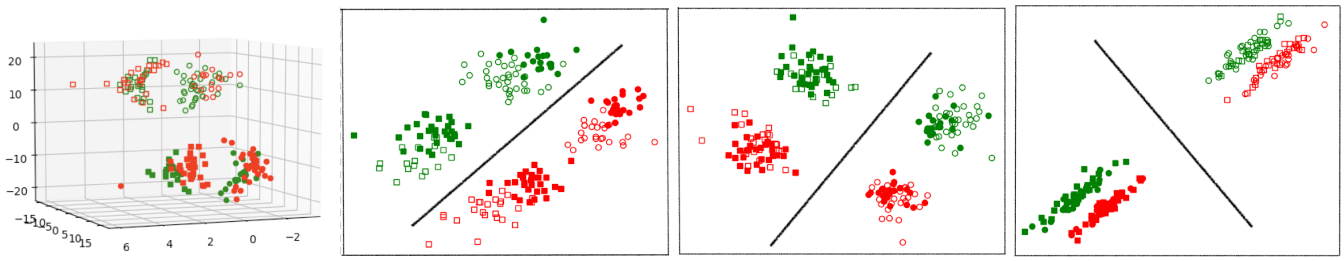

Fig. 6: The output of Algorithm 2 on the Clustering dataset. The first subfigure shows the 3D embedding and the remaining ones correspond to the 3 projections. To make it easier to see the 3 different types of clusters captured here, in all subfigures we use green/red to distinguish between datapoints from different clusters of the first type, circle/square glyphs for clusters of the second type, and filled/empty glyphs for clusters of the third type. We manually add the separating lines to highlight the separations realized in the 3 planes. The final MPSE stress for the Clustering dataset with varying projections (Algorithm 2) is 0.60, with projection-wise stress values of 0.52, 0.47 and 0.77.

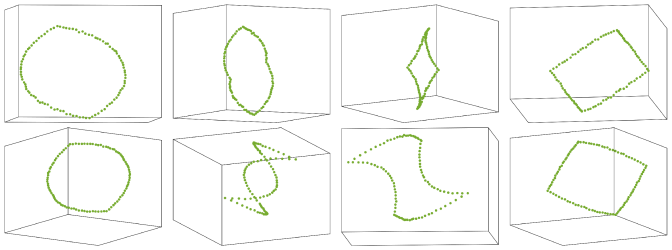

Fig. 7: The output of Algorithms 1 and 2 applied on the Circle-Square dataset. The first row show the result of Algorithm 1 and the second row shows the result of Algorithm 2. For both rows, the first column captures the recovered circle and the last row captures the recovered square. The second and third rows show the rotational transition in 3D between the circle and the square, revealing its 3D structure. The final MPSE stress for the circle-square dataset with fixed projections (Algorithm 1)is 0.046, with projection-wise stress values of 0.045 and 0.047. The final MPSE stress for the circle-square dataset with varying projections (Algorithm 2) is 0.044, with projection-wise stress values of 0.044 and 0.045.

constant $c = 0.01$ with smart initialization described in Algorithm 3. The results are shown in the second row of Fig. 7. As we can see the algorithm again successfully placed the points in 3D so that one can see a circle and a square from different directions (first and last subfigures).

4.3 Clusters Dataset

One of the many applications of dimensionality reduction is to preprocess the dataset by reducing its dimension and then apply a clustering/classification algorithm. The setting where we want to test whether our proposed algorithm preserves the clusters of a given dataset is the following: We want to visualize a dataset in 3D such that different 2D projections capture different clusters (say, determined by considering different attributes of the data). For this purpose, we fix $n = 200$ and create 3 datasets in 2D each having 2 well separated clusters of 100 datapoints each. Our goal is to apply Algorithms 1 and 2 and check whether the corresponding 3D embeddings preserve the clusters and if so what the corresponding projections look like. We apply Algorithm 2 with $T = 100$, the distance matrices corresponding to datapoints of subfigures of Fig. 8 and random initialization. The results are shown in Fig. 6.

Remarkably, as shown in Fig. 6, the MPSE algorithm with varying projections captured the special structure of the dataset (three different pairs of clusters defined on the same datapoints) and embedded them in 3D so that the 3 different separations can be seen from the corresponding projections.

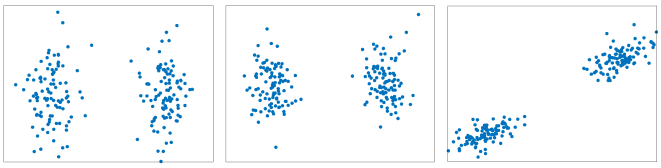

Fig. 8: The input to the Clusters dataset: each subfigure shows an example of a dataset of 200 points with 2 well separated clusters of 100 points.

4.4 15th-century Florentine Families Dataset

We now consider a network dataset that captures social relationships between prominent families in Renaissance Florence (one of the largest and richest cities in 15th century Europe) [40, 44]. This dataset contains descriptions of social ties between Florentine families during the years 1426-1434, a period of historical significance that marks the rise to power of the Medici family.

The social ties are divided into categories such as 'marriage' (number of marriages between two families) and 'loan' (number of loans between two families). We can interpret this dataset as a network with multiple attributes. To visualize this network, we select a subset of the families such that the network structure for the 'marriage' (1) and 'loan' (2) attributes each form a connected graph. For each type of relation $k \in \{1, 2\}$, we define the dissimilarity between two families as follows: for families i and j, if there is a nonzero number $n_{ij}^{(k)}$ of ties between the two families, then the pairwise dissimilarity is initially given by $D_{ij}^{(k)} = 1/n_{ij}^{(k)}$; afterwards, shortest path distances are computed for all pairs of families using this initial set of dissimilarities, resulting in dissimilarities for every pair of families. The pairwise weights used to produce the MDS (7) and MPSE (8) embeddings are defined by $w_{ij}^{(k)} = 1/D_{ij}^{(k)}$.

We compute a 2D MDS embedding based on marriage distances and a 2D MDS embedding based on loan distances (independent of each other). We then compute a 3D simultaneous embedding based on both marriage and loan distances using MPSE with variable projections with the following parameters: maximum number of iterations $T = 300$, starting learning rate $\mu_0 = 1$, stochastic constant $c = 0.1$, and initial embedding from Algorithm 3.

The first row of images in Fig. 10 shows the two 2D MDS embeddings and the second row of images shows two projections of the 3D MPSE embedding. The first column corresponds to marriages and the second column corresponds to loans. The stress value for each embedding is the standard normalized

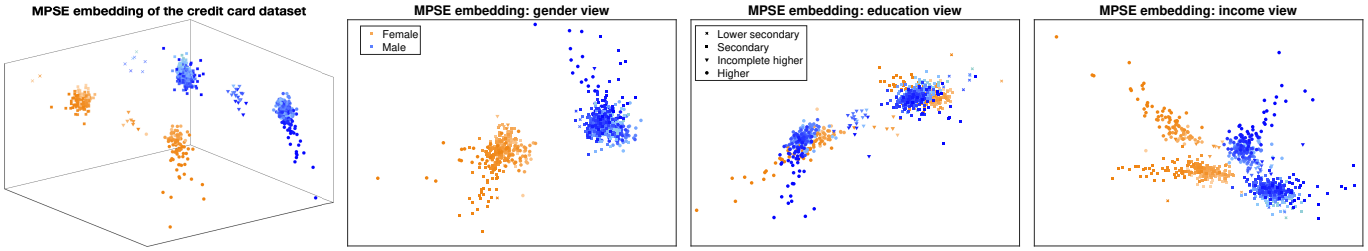

Fig. 9: The output of Algorithm 2 on the credit card dataset. The first subfigure shows the 3D embedding and the remaining ones correspond to the 3 projections on gender, education and income views, respectively. In all figures, high color intensity represents high income, color represents gender, and glyph shapes represent education levels. The final MPSE stress for the credit card dataset with varying projections is 0.40, with projection-wise stress values of 0.28, 0.29 and 0.55.

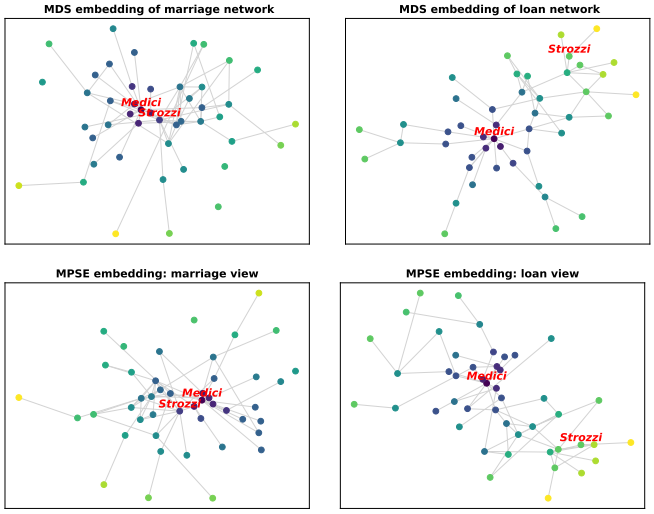

Fig. 10: MDS and MPSE embeddings of the Florentine families dataset based on marriages and loans. The first row shows the two 2D MDS embeddings (obtained independently of each other) and the second row shows two projections of the 3D MPSE embedding from Fig. 2. The color indicates the true graph distance from the Medici family for each attribute with blue indicating close neighbors and yellow indicating distant ones (as described in the text). The MDS stress values are 0.70 for the marriage distances and 0.65 for the loan distances. The total MPSE stress is 0.78, with projection-wise stress values of 0.75 for marriage perspective and 0.81 for the loan perspective.

MDS stress (7). Hence, the MDS and MPSE values are directly comparable to each other. Note that the individual normalized MDS stress for both perspectives in the MPSE embedding are close to the optimal normalized MDS stress values (computed individually for each attribute). This indicates that it is possible to visualize both attributes simultaneously with the MPSE method, without significantly increasing stress values.

The colors in Fig. 10 indicate the true graph distance from the Medici family, with respect to marriage relations (the first column) and loan relations (the second column). A "perfect" embedding would result in dark blue colors near the Medici family and yellow for the nodes farthest away (with gradual blue-to-yellow transition in between). Note that we can obeserve this phenomenon both in the individual MDS embeddings and in the MPSE embeddings in Fig. 10. Different views of the 3D MPSE embedding are shown in Fig. 2, with the view corresponding to the optimal MPSE perspective of the marriage attribute on the left and the view corresponding to the optimal MPSE perspective of the loan attribute on the right.

A bit of historical background can help us see some interesting results in our visualizations. The Strozzi family had been Florence's richest one, but ended up exiled and in ruin after the Medici took control of Florence's government in 1434. The MPSE embedding, illustrated in Fig. 2 and Fig. 10, allows us a glimpse in this story. The Medici family occupies a central location in both the individual MDS and in the MPSE embeddings. The Strozzi family occupies a similarly central position in the marriage perspectives, but not in the loan perspectives. This reflects the nature of their political power at the time, as both families has strong ties to other prominent families in Florence, but the Medici family was doing better financially by that time. The better positioning of the Medici family is nicely illustrated in the MPSE 3D embedding in Fig. 2 in a way that cannot be illustrated by the MDS embeddings: the Medici family has a central position in the 3D embedding (with an average pairwise distance of 2.87), whereas the Strozzi family lies closer to the periphery (with an average pairwise distance of 4.05). Even though their position in the marriage network was firm, their position in the overall network was weaker.

4.5 Credit Card Application Dataset

Another real-world, high-dimensional dataset comes from anonymized credit card applications [1]. The dataset contains information about customers of a bank (gender, education level, income, etc.). We randomly selected 1000 customers and focused on three parameters: annual income, gender, and level of education. The goal is to use MPSE to embed the dataset in 3D so that three different projections show us patterns based on the corresponding parameters. Since MPSE requires numeric input we modify some of the parameters a bit. For gender we map male and female to 0 and 1. For education level, we map lower secondary, secondary, incomplete higher and higher education to 0, 1, 2, 3. To avoid giving more importance to some of the features, each set of features is then normalized so that the scale of all MPSE perspectives is the same.

We compute an MPSE variable projections embedding (Algorithm 2), after initialing it with Algorithm 3, using the following parameters: maximum number of iterations $T = 200$, starting learning rate $\mu_0 = 1$, and stochastic constant $c = 0.004$. The results are shown in Fig. 9. The first subfigure shows one view of the resulting 3D embedding, and the remaining subfigures show the projections of the embedded dataset onto the gender, education level, and income views.

The 3D embedding seems to capture well the different perspectives, shown in the 3 different projections. The gender view shows clearly separated clusters of male (orange) and female (blue color) customers. The education view groups customers with similar education levels (see the groups of squares, circles, and triangles). The income view attempts to "sort" the customers by income (high on the top, low on the bottom in this figure), as captured in the change of color intensity from the top to the bottom. This view also shows us that educa-

tion level and income are correlated (circles at the top), and that men have higher income than women (the orange clusters are higher than the blue clusters). Indeed the actual averages for male and female customers in this sample of size 1000 are $195000 and $152000, respectively.

5 Testing the scalability of Algorithms 1 and 2

In this section we test whether Algorithms 1 and 2 scale once the number of datapoints and the number of projections increase. For this purpose we create random datasets, that is we fix the number of datapoints n and sample n points uniformly from a solid ball with radius 1 in 3D. Next, we fix the number number of projections K and generate K random projections $\mathbf{Q}^1, \ldots, \mathbf{Q}^K$ of this data onto $\mathbb{R}^2$. From these projected datasets we generate the corresponding distance matrices $\mathbf{D}^1, \ldots, \mathbf{D}^K$. The input for Algorithm 1 is $\mathbf{D}^1, \ldots, \mathbf{D}^K$ and $\mathbf{Q}^1, \ldots, \mathbf{Q}^K$ and for Algorithm 2 the input is $\mathbf{D}^1, \ldots, \mathbf{D}^K$. We consider the following 2 experiments for Algorithm 1 and Algorithm 2: The first experiment fixes the number of projections to be $K = 3$ and varies the number of datapoints n between 100 and 2000 in increments of 100. We fix the max number of iterations $T = 100$ for the first experiment and compute the average time, and the number of times that the algorithm successfully finds the global minimum over 10 instances.

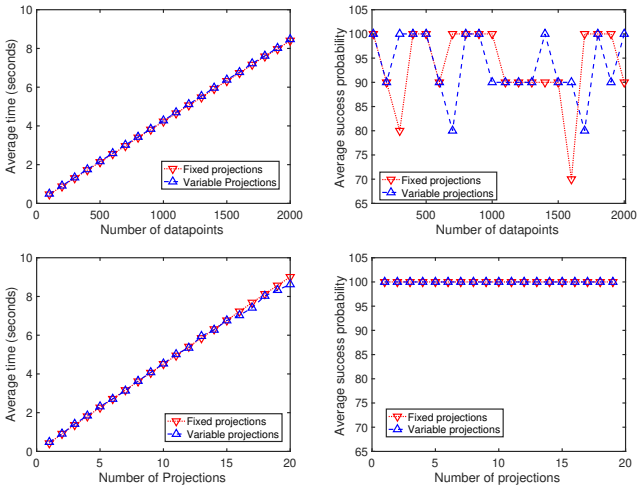

Fig. 11: Summary of the results for the scalability tests. The first row corresponds to the experiments where we fix the number of projections $K = 3$ and vary number of datapoints between 200 to 2000 with increments of 100. The second row corresponds to the experiments where we fix the number of datapoints $n = 200$ and vary the number of projections K from 1 to 19 with increments of 1. The first column reports the average running time of the algorithms over 10 instances and the second row shows the success rates of the algorithms.

The second experiment fixes $n = 200$ and varies the number of projections K between 1 and 20 in increments of 1. We fix the max number of iterations $T = 1000$ and compute the average time and number of times that the algorithm successfully finds the global minumum over 10 instances. The results are reported in Fig. 11. We remark that, as shown in Fig. 11, the running time of Algorithms 1 and 2 increases linearly with the increase of datapoints with Algorithms 2. Similarly, the running time of both algorithms increases linearly with the increase of the number of projections. The second column of Fig. 11 shows that the increase in the number of datapoints and the increase of number of projections does not effect the success rate of both algorithms.

Fig. 12 shows the computation history for Algorithm 1 when applied to the "1, 2, 3" dataset which resulted in the images shown in Fig. 1. The left image shows multi-perspective MDS stress at each iteration. The right image shows the behavior of parameters (normalized gradient size, learning rate, and normalized step size) during the execution of the algorithm.

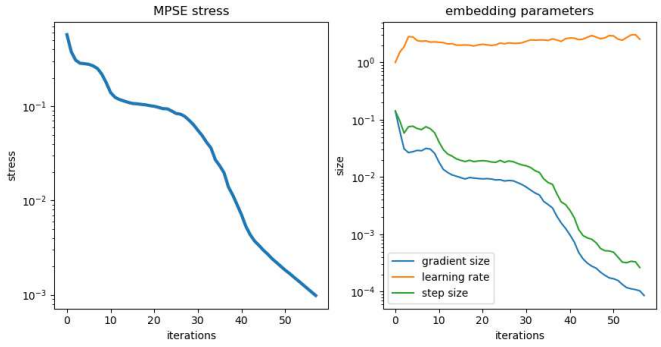

Fig. 12: Computation history of Algorithm 1 for the "1, 2, 3" data shown in Fig. 1. The left image shows the decay of total multi-perspective MDS stress at each iteration. The right image shows the behavior of parameters (normalized gradient size, learning rate, and normalized step size) during the execution of the algorithm. We used the following parameters for Algorithm 1: initial learning rate $\mu_0 = 1$, stochastic constant $c = 0.02$, and random initial embedding. The final MPSE stress of 0.001 was reached after 58 iterations, with projection-wise MDS stress values of 0.0009, 0.0008 and 0.0011.

6 Conclusions and Future Work

We described a generalization of MDS which can be used to simultaneously optimize multiple distance functions defined on the same set of objects. The result is an embedding in 3D space with a set of given or computed projections that show the different views. This approach has applications for visualizing abstract data and multivariate networks. It would be interesting to experiment and test whether MPSE can compete with current techniques for 3D reconstruction from 2D images. For example, one could combine the MPSE approach with a multiway matching algorithm [22, 39] and a feature extraction algorithm [33] to automatically detect points of interests in each image, match them, and place them in 3D to recover the 3D figure. The MPSE webpage http://mpse.arl.arizona.edu provides 3D visualizations with interactive examples, and a functional MPSE prototype; https://github.com/enggiqbal/MPSE provides the source code.

Acknowledgements

This research was supported in part by National Science Foundation grants CCF–1740858, CCF-1712119, and DMS-1839274.

We thank the participants in the "Visualizing Dynamic Graphs in VR" working group at Shonan Seminar 131 "Immersive Analytics for Network and Trail Sets Data Analysis": David Auber, Peter Eades, Seokhee Hong, Hiroshi Hosobe, Stephen Kobourov, Kwan-Liu Ma, and Ken Wakita. We also thank the participants in the "Linked Networks Perspectives for Humanities" working group at Dagstuhl Seminar 18482 "Network Visualization in the Humanities": Gregor Betz, Stephen Kobourov, Martin Nöllenburg, Gerik Scheuermann, Timothy Tangherlini, and Christopher Warren.

References

[1] Credit Card Fraud Detection. https://www.kaggle.com/c/home-credit-default-risk. Accessed: 2020-07-20.

[2] P. Angelini, M. Geyer, M. Kaufmann, and D. Neuwirth. On a tree and a path with no geometric simultaneous embedding. In *International Symposium on Graph Drawing*, pp. 38–49. Springer, 2010.

[3] D. Auber, D. Archambault, R. Bourqui, M. Delest, J. Dubois, A. Lambert, P. Mary, M. Mathiaut, G. Mélançon, B. Pinaud, et al. Tulip 5, 2017.

[4] S. Bai, X. Bai, L. J. Latecki, and Q. Tian. Multidimensional scaling on multiple input distance matrices. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[5] M. Bastian, S. Heymann, M. Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *ICWSM*, 8:361–362, 2009.

[6] V. Batagelj and A. Mrvar. Pajek-program for large network analysis. *Connections*, 21(2):47–57, 1998.

[7] A. Bezerianos, F. Chevalier, P. Dragicevic, N. Elmqvist, and J.-D. Fekete. Graphdice: A system for exploring multivariate social networks. In *Computer Graphics Forum*, vol. 29, pp. 863–872. Wiley Online Library, 2010.

[8] T. Bläsius, S. G. Kobourov, and I. Rutter. Simultaneous embedding of planar graphs. In R. Tamassia, ed., *Handbook of Graph Drawing and Visualization*, pp. 349–381. CRC Press, 2013.

[9] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

[10] P. Brass, E. Cenek, C. A. Duncan, A. Efrat, C. Erten, D. P. Ismailescu, S. G. Kobourov, A. Lubiw, and J. S. Mitchell. On simultaneous planar graph embeddings. *Computational Geometry*, 36(2):117–130, 2007.

[11] L. Chen and A. Buja. Local multidimensional scaling for nonlinear dimension reduction, graph drawing, and proximity analysis. *Journal of the American Statistical Association*, 104(485):209–219, 2009.

[12] I. F. Cruz and J. P. Twarog. 3d graph drawing with simulated annealing. In *International Symposium on Graph Drawing*, pp. 162–165. Springer, 1995.

[13] E. Di Giacomo, W. Didimo, M. J. van Kreveld, G. Liotta, and B. Speckmann. Matched drawings of planar graphs. *J. Graph Algorithms Appl.*, 13(3):423–445, 2009.

[14] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz - open source graph drawing tools. In *Graph Drawing*, pp. 483–484, 2001.

[15] N. Elmqvist, P. Dragicevic, and J.-D. Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE transactions on Visualization and Computer Graphics*, 14(6):1539–1148, 2008.

[16] J. J. Fowler, M. Jünger, S. G. Kobourov, and M. Schulz. Characterizations of restricted pairs of planar graphs allowing simultaneous embedding with fixed edges. *Computational Geometry*, 44(8):385–398, 2011.

[17] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In *International Symposium on Graph Drawing*, pp. 239–250. Springer, 2004.

[18] M. Geyer, M. Kaufmann, and I. Vrto. Two trees which are self–intersecting when drawn simultaneously. In *International Symposium on Graph Drawing*, pp. 201–210. Springer, 2005.

[19] M. Ghoniem, F. Mcgee, G. Melançon, B. Otjacques, and B. Pinaud. The state of the art in multilayer network visualization. *arXiv preprint arXiv:1902.06815*, 2019.

[20] L. Grilli, S.-H. Hong, G. Liotta, H. Meijer, and S. K. Wismath. Matched drawability of graph pairs and of graph triples. *Computational Geometry*, 43(6-7):611–634, 2010.

[21] M. I. Hossain, V. Huroyan, S. Kobourov, and R. Navarrete. Multi-perspective, simultaneous embedding. *arXiv preprint arXiv:1909.06485*, 2019.

[22] V. Huroyan. *Mathematical Formulations, Algorithms and Theory for Big Data Problems*. PhD thesis, University of Minnesota, 2018.

[23] I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 1986. doi: 10.1007/978-1-4757-1904-8

[24] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inform. Process. Lett.*, 31:7–15, 1989.

[25] S. Kanaan-Izquierdo, A. Ziyatdinov, M. A. Burgueño, and A. Perera-Lluna. Multiview: a software package for multiview pattern recognition methods. *Bioinform.*, 35(16):2877–2879, 2019.

[26] A. Kerren, H. C. Purchase, and M. O. Ward. Introduction to multivariate network visualization. In *Multivariate Network Visualization*, pp. 1–9. Springer, 2014.

[27] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of complex networks*, 2(3):203–271, 2014.

[28] S. G. Kobourov. Force-directed drawing algorithms. In R. Tamassia, ed., *Handbook of Graph Drawing and Visualization*, pp. 383–408. CRC Press, 2013.

[29] S. G. Kobourov and K. Wampler. Non-eeuclidean spring embedders. *IEEE Transactions on Visualization and Computer Graphics*, 11(6):757–767, 2005.

[30] Y. Koren and L. Carmel. Visualization of labeled data using linear transformations. In *IEEE Symposium on Information Visualization 2003*, pp. 121–128. IEEE, 2003.

[31] A. Koutsoudis, B. Vidmar, G. Ioannakis, F. Arnaoutoglou, G. Pavlidis, and C. Chamzas. Multi-image 3d reconstruction data evaluation. *Journal of Cultural Heritage*, 15(1):73–79, 2014.

[32] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[33] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision, Kerkyra, Corfu, Greece, September 20-25, 1999*, pp. 1150–1157. IEEE Computer Society, 1999. doi: 10.1109/ICCV.1999.790410

[34] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11):2579–2605, 2008.

[35] Y. Malitsky and K. Mishchenko. Adaptive gradient descent without descent. *CoRR*, abs/1910.09529, 2019.

[36] L. McInnes, J. Healy, N. Saul, and L. Großberger. UMAP: uniform manifold approximation and projection. *J. Open Source Softw.*, 3(29):861, 2018. doi: 10.21105/joss.00861

[37] T. Munzner. Exploring large graphs in 3d hyperbolic space. *IEEE Computer Graphics and Applications*, 18(4):18–23, 1998.

[38] O. Özyesil, V. Voroninski, R. Basri, and A. Singer. A survey of structure from motion. *Acta Numer.*, 26:305–364, 2017. doi: 10.1017/S096249291700006X

[39] D. Pachauri, R. Kondor, and V. Singh. Solving the multiway matching problem by permutation synchronization. In *Advances in neural information processing systems*, pp. 1860–1868, 2013.

[40] J. Padgett and C. Ansell. Robust action and the rise of the medici, 1400-1434. *American Journal of Sociology*, 98:1259–1319, 1993.

[41] C. Pich. *Applications of multidimensional scaling to graph drawing*. PhD thesis, 2009.

[42] A. J. Pretorius and J. J. Van Wijk. Visual inspection of multivariate graphs. In *Computer Graphics Forum*, vol. 27, pp. 967–974. Wiley Online Library, 2008.

[43] Y. Quéau, J. Mélou, J.-D. Durou, and D. Cremers. Dense multiview 3d-reconstruction without dense correspondences. *arXiv preprint arXiv:1704.00337*, 2017.

[44] T. Shafie. A multigraph approach to social network analysis. *Journal of Social Structure*, 16, 2015.

[45] R. N. Shepard. The analysis of proximities: multidimensional scaling with an unknown distance function. *Psychometrika*, 27(2):125–140, 1962.

[46] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE transactions on visualization and computer graphics*, 12(5):733–740, 2006.

[47] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *International journal of computer vision*, 80(2):189–210, 2008.

[48] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. W. Fellner. Visual analysis of large graphs: state-of-the-art and future research challenges. In *Computer graphics forum*, vol. 30, pp. 1719–1749. Wiley Online Library, 2011.

[49] Y. Wang, Y. Wang, Y. Sun, L. Zhu, K. Lu, C.-W. Fu, M. Sedlmair, O. Deussen, and B. Chen. Revisiting stress majorization as a unified framework for interactive constrained graph visualization. *IEEE transactions on visualization and computer graphics*, 24(1):489–499, 2018.

[50] M. Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pp. 811–819. ACM, 2006.

[51] J. X. Zheng, S. Pawar, and D. F. Goodman. Graph drawing by stochastic gradient descent. *IEEE transactions on visualization and computer graphics*, 25(9):2738–2748, 2018.