Multi-Perspective, Simultaneous Embedding

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

1 Introduction

Typically, when given a high dimensional dataset, the goal of visualizing it often is interpreted as finding some 2D placement of the objects such that similar objects are close to each other and dissimilar ones are far. Similarly, given a graph $G = (V,E)$ the goal of visualizing it can be summed up as finding some 2D layout that represents the underlying structure. Dating back to the 1960s, a classical tool that is used for both graphs and high dimensional dataset visualization, is Multidimensional Scaling (MDS) which aims to preserve the distances between all pairs of nodes/objects [33]. More recent dimensionality reduction approaches, such as t-SNE [28], aim to preserve local neighborhoods and clustering.

Now consider a more general case when the input is 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 department). We would like to compute a layout $L$ in 3D as well as 3 planes such that when $L$ 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 the high dimensional setting, the goal is to perform dimensionality reduction to 3D, given a set of objects, several pairwise distance functions between them, and the same number of projection planes. The optimization goal now is to simultaneously preserve the distances between the objects when projected to the corresponding planes.

In both settings, this is a strict generalization of the underlying classical problem, which can be seen as a special case when only one pairwise distance function is given. Even this special case is known to be difficult as the standard optimization approaches such as gradient descent do not necessarily converge to the global optimum. Nevertheless, in practice, when there is 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 MPSE produces good solutions.

We describe the MPSE method in detail and also briefly mention how it is implemented. 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. We illustrate performance with several examples.

A common approach for visualizing different relationships on the same set of objects involve 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 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. This becomes more plausible in 3D, and with the advent of virtual reality and augmented reality systems, 3D visualization and interaction itself is becoming a reality. Still, when presenting 3D results in a paper we are limited to showing 2D snapshots. We include 3D visualizations with interactive examples on this webpage https://uamap-dev.arl.arizona.edu/static/MPSE/index.html

1.1 Previous work

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

Multivariate network visualization. Multivariate [21] and multilayer [17] 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. [22]. Wattenberg’s PivotGraph [38] 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 [34] unfold multiple attributes of a graph, a pair of attributes at a time, using two dimensions. Pretorius and van Wijk [31] 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 [6] 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. When different attributes are selected, the matrix of node-link diagrams morphs from the old to the new. This system is built on the earlier ScatterDice system [14].

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 should help 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 [23]. Visual analytics systems for graphs usually focus on interaction [36]. MDS-like approaches to drawing graphs are exemplified in algorithms such as that of Kamada-Kawai [19], Koren and Carmel [25]. Most commonly used graph drawing systems, such as Graphviz [13], pajek [5], Tulip [2] and Gephi [4], provide options to visualize graphs in 3D based on MDS-like optimization. Variants of MDS are used in many graph layout systems, including [10, 15, 30, 37]. Other approaches to exploring layouts in 3D include 3D hyperbolic and spherical spaces [11, 24, 29].

Multidimensional scaling. Multidimensional scaling (MDS) is a well known dimensionality reduction and data visualization technique. The problem was first studied in the non-metric setting by Shepard [33] and Kruskal [27]. 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. The metric variant of MDS is more frequently used and it relies on the exact distances. The goal of metric MDS is to place objects in some low dimensional space so as to preserve the given pairwise distances between the objects. Given a distance matrix (pairwise dissimilarity matrix) \( D = (d_{ij})_{n \times n} \), between \( n \) objects, the objective function function for MDS is

\[
S(x_1, \ldots, x_n) = \sum_{i \neq j} (d_{ij} - \|x_i - x_j\|^2)^2
\]  

(1)

The function defined in (1) is called the stress function. Some well known techniques for minimizing the stress function (1) are standard gradient descent, stochastic gradient descent [8], and stress majorization [15].

In their Multiview package, Kanaan et al. [20] provide algorithms to embed data in Euclidean space by combining multiple distance matrices between the objects in the data. This is different from what we do as we are simultaneously visualizing multiple distance matrices and make is possible to see the different relationships by changing the viewpoints. Similarly, Bai et al. [3] combine multiple distance matrices in a generalization of multidimensional scaling. Again, this is different from what we do as we are simultaneously visualizing multiple distance matrices and make is possible to see the different relationships by changing the viewpoints.

Simultaneous embedding. This problem is also related to simultaneous graph embedding and matched drawings of graphs [7]. 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 and solutions are guaranteed to exist for very restricted type of input graphs, such as two paths [9], while instances with no solutions can be constructed from a pair of trees [16] or even a (path, tree) pair [1]. Matched drawings require straight-line drawings of the two or more input graph 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 [18]. In general, instances with no solution can be constructed from a pair of planar graphs, or even a (planar graph, tree) pair [12]. 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.

3D Reconstruction. Our problem is also related to 3D reconstruction problem 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 [35]. More restricted variants are even closer to our setting [26, 32]. 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.

1.2 Our Contribution

The main contribution in this paper is a generalization of MDS to multiple distance matrices. 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.

### 1.3 Structure of This Paper

The rest of this paper is organized as follows: Section 2 gives the mathematical definition of the MPSE problem and includes a solution for MPSE with fixed projections and an algorithm for MPSE with varying projections; Section 3 demonstrates numerical experiments for both of the proposed algorithms; and Section 4 includes the mathematical background needed for the proposed algorithms.

## 2 Multiview Graph Visualization

We begin this section with a brief review of the standard MDS problem and then an overview of our multiview-MDS formulation. Let $d$ be an $n$ by $n$ matrix containing pairwise dissimilarity measures between $n$ objects. The goal in multidimensional analysis is to assign positions $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ to the $n$ objects so that the resulting pairwise distances $||x_i - x_j||$ are as close as possible to the observed pairwise dissimilarities $d_{ij}$, as measured by the MDS stress function (1). If the minimum of the MDS stress function is zero, then the objects can be positioned so that their pairwise distances exactly represent the pairwise dissimilarities $d$. If the minimum of the MDS stress function is greater than zero, a minimizer of (1) still provides an approximate way to visualize the dissimilarities.

Suppose that instead of a single pairwise dissimilarity matrix $D$, we observe multiple pairwise dissimilarity matrices $D^1, D^2, \ldots, D^k$ for the same set of $n$ objects. It is natural to ask if an embedding $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ of the objects exists so that the different dissimilarities can be visualized by the relative positions of $x_1, x_2, \ldots, x_n$, but there is no way to accomplish this without further assumptions about the relationship between the different pairwise dissimilarity matrices.

Motivated by the problem of 3D reconstruction from multiple 2D images, we consider the following question: is it possible to place the $n$ objects under consideration in 3D, $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$, so that the different pairwise dissimilarity matrices $D^1, D^2, \ldots, D^k$ are recovered from the pairwise dissimilarity matrices between the objects after being projected to different 2-dimensional subspaces of $\mathbb{R}^3$? For example, the pairwise dissimilarity matrices $D^1, D^2$, and $D^3$ could be given by $D^1_{ij} = ||x_i - x_j||$, $1 \leq i \leq 3$, and $D^2_{ij} = ||x_i - x_j||$, $1 \leq i \leq 3$, and $D^3_{ij} = ||x_i - x_j||$, $1 \leq i \leq 3$. We observe the 3 by 3 orthogonal projection matrices that project onto the $xy$ plane, $xz$ plane, and $yz$ coordinate planes. In this scenario, we can ask if the positions $x_1, x_2, \ldots, x_n$ can be recovered from the pairwise dissimilarity matrices, assuming that the corresponding projections are known, or if both the positions and projections can be recovered from the pairwise dissimilarity matrices alone. Even for sets of pairwise dissimilarity measures that are not generated this way, this assumption can be used to form visualizations that can simultaneously illustrate the different dissimilarity measures.

### 2.1 MPSE with Fixed Projections

We consider the following problem. We have $n$ objects that we wish to embed in 3D space. We begin with three pairwise dissimilarity matrices $D^1, D^2, D^3$, along with the corresponding orthogonal projection matrices $\Pi_1, \Pi_2, \Pi_3$. We wish to find the positions $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ that best agree with the set of distance/projection pairs. For this purpose, we define the multiview-MDS stress function:

$$S_M(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{3} \sum_{j<i} \left( D^i_{ij} - \|\Pi_i x_i - \Pi_j x_j\|_2 \right)^2$$

This function measures disagreement between each of the pairwise dissimilarity matrices and the pairwise distances between the corresponding projected positions. The goal is to find a set of positions that minimize the multiview-MDS stress function (2), that is

$$\min \ S_M(x_1, x_2, \ldots, x_n)$$

In our discussion and experiment, the matrices $\Pi_1, \Pi_2$ and $\Pi_3$ are always orthogonal projection matrices of rank 2. A $3 \times 3$ matrix $\Pi$ is an orthogonal projection matrix if $\Pi^2 = \Pi = \Pi^T$. For example, a natural triple is to fix the projection matrices as the projections onto the $xy$, $xz$ and $yz$ planes, as given by

$$E_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, E_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, E_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

We will refer to these as the “unit cube projections”. Another set of fixed projection matrices that we use in our experiments are projections onto the $xz$ plane and the two planes that form $\pi/3$ and $2\pi/3$ angles with $xz$ plane (corresponding to a rotation of the 3D input around the z-axis):

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & \sqrt{3} & 0 \\ 0 & 0 & 0 \\ 0 & \sqrt{3} & 0 \end{bmatrix}, \begin{bmatrix} 1 & -\sqrt{3} & 0 \\ 0 & 0 & 0 \\ 0 & \sqrt{3} & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Problem (3) can be formulated for any number of distance matrices and corresponding projections, and the dimension of the ambient space and rank of the projections can be arbitrary. The restrictions on $\Pi_1, \Pi_2$ and $\Pi_3$ could also be removed to allow for more general linear or even non-linear maps. For clarity, we limit our presentation to the setup with 3 distance matrices and 3 orthogonal projections.

An important property of the objective function for multiview MDS in (1) is that it is differentiable. We solve (3) with gradient descent. Since the objective function defined in (2) is not convex, one needs to be careful when choosing the initial configuration and learning rate (the size of the movement along the gradient) in order for the algorithm to converge to a local minimum. Thus, to find the optimal point an appropriate initialization and learning rate are required. We summarize the gradient descent algorithm for the multiview MDS in Algorithm 1.

**Algorithm 1 Gradient Descent Algorithm for MPSE**

**Input:** Distance matrices: $D^1, D^2, D^3 \in \mathbb{R}^{n \times n}$, learning rate: $\mu$, maximum number of iterations: $N$

**Initialize** $X^{(0)} = [x_1^0, x_2^0, \ldots, x_n^0] \in \mathbb{R}^{3 \times n}$

**while** $i < N$ **do**

$X^{(i+1)} = X^{(i)} - \mu \nabla S_M(X^{(i)})$

**end while**

**Output:** $x_1^N, x_2^N, \ldots, x_n^N \in \mathbb{R}^3$

### 2.2 MPSE with Varying Projections

In this section, we again consider minimization of the multiview-MDS stress function (2), but we no longer assume that the projection matrices are given. Our goal is then to find both the positions $x_1, x_2, \ldots, x_n \in \mathbb{R}^3$ and the projection matrices $\Pi_1, \Pi_2$ and $\Pi_3$ that best capture the given distance matrices $D^1, D^2$ and $D^3$.

We now formulate the MPSE problem for varying projections. As before, we assume that $D^1, D^2, D^3$ are the distance matrices observed after projecting by $\Pi_1, \Pi_2$ and $\Pi_3$, respectively, but these projection matrices are no longer known. Note that a $3 \times 3$ matrix $\Pi$ is a rank-2 orthogonal projection matrix if and only if $\Pi = QQ^T$, where $Q$ is an orthonormal $3 \times 2$ matrix. Let $\Omega^{3 \times 2}$ be the set of all orthogonal $3 \times 2$ matrices and let $X = [x_1, x_2, \ldots, x_n]$. The multiview MDS stress function...
Algorithm 3.\textit{A}ction of \textit{pute} \textit{projected} gradient descent, where \textit{the} algorithm \textit{alternates} between mini-
\textit{mizing} (6) \textit{with} respect to \textit{the} \textit{matrix} \textit{X} and \textit{each} of the \textit{pro}-
\textit{jections}. \textit{At any given iteration, in order to update the positions}
\textit{X}, \textit{the current projection matrices} are \textit{fixed} and \textit{Algorithm 1}
is \textit{used}. \textit{Then, we fix X and minimize} (6) \textit{for} each of the \textit{pro}-
\textit{jection matrices} \textit{separately}. \textit{The procedure} is \textit{repeated} until 
\textit{convergence}. \textit{Since} \textit{convergence} is not \textit{guaranteed}, the \textit{algo}-
\textit{rithm} \textit{may} be \textit{terminated} when a \textit{fixed number} of \textit{iterations} is 
\textit{reached}. \textit{This algorithm} is \textit{summarized} in \textit{Algorithm 2}. \textit{Details}
on \textit{each of the steps} are \textit{presented} \textit{afterwards}.

\textbf{Algorithm 2} MPSE with Unknown Projections

\textbf{Input:} Distance matrices: $D^1, D^2, D^3$, learning rates: $\mu_X$ and $\mu_Q$, number of initial iterations: $N_X$ and $N_Q$, number of loops: $M$, number of iterations per loop: $M_X$ and $M_Q$.

\textbf{Compute $X^{(0)}$ and $Q_1^{(0)}, Q_2^{(0)}$ and $Q_3^{(0)}$ using Algorithm 3}.

\textbf{while} $i \leq M$ \textbf{do}

\hspace{1em} \textbf{Compute $X^{(i+1)}$ using Algorithm 1} with initial positions $X^{(i)}$ and fixed projections $Q_1^{(i)}, Q_2^{(i)}$ and $Q_3^{(i)}$.

\hspace{1em} \textbf{For $l \in \{1, 2, 3\}$, compute $Q_l^{(i+1)}$ using Algorithm 3 with initial orthogonal matrix $Q_l^{(i)}$ and fixed positions $X^{(i+1)}$.}

\textbf{end while}

\textbf{Output:} $X^{(M)}$ and $Q_1^{(M)}, Q_2^{(M)}$ and $Q_3^{(M)}$.

\text{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 $P(A) \in \mathbb{O}^{3 \times 2}$ that minimizes $\|A - Q\|_F$ among all $Q \in \mathbb{O}^{3 \times 2}$. There is a simple way to compute $P(A)$: if $USV^T$ is the reduced singular value decomposition of $A$, then $P(A) = UV^T$. The algorithm is summarized in Algorithm 3.

\textbf{Algorithm 3} Projected gradient descent for MPSE

\textbf{Input:} Distance matrix: $D$, fixed positions: $X$, initial orthogonal matrix: $Q^{(0)}$, learning rate: $\mu$, number of iterations: $N$.

\textbf{while} $i \leq N$ \textbf{do}

\hspace{1em} $\bar{Q}^{(i+1)} = Q^{(i)} - \mu \nabla Q S_M(X; Q^{(i)})$

\hspace{1em} $\tilde{Q}^{(i+1)} = P(\bar{Q}^{(i+1)})$

\textbf{end while}

\textbf{Output:} $Q^{(N)}$.
with smaller weights are put close to each other and vertices
and wants to visualize it in 2D or 3D such that the vertices
problems. That is, if one has a weighted/unweighted graph
amples. However, MDS is also used for graph visualization
The previous three examples were data/shape visualisation ex-
3.4 Grid-Path graph Dataset Description
The next dataset that we create consists of the geometric
shapes of a circle, square and triangle. We refer to this dataset
the cluster information is well preserved in all three projections.
We present the results of the application of Algorithm 1 on
this dataset in Figure 5. For the first row of Figure 5 we in-
put only the datasets for circle and square. We note, that
Algorithm 1 captures both shapes very well. The second row
of Figure 5 presents the results of Algorithm 1 for the input
dataset that contains square, circle and triangle with projec-
tions from (4). Note that the shapes are not perfect, but the
algorithm recovered the circle in the first figure, square in the
second one and triangle in the third one.
3.3 Clusters Dataset Description
One of the many applications of dimensionality reduction is to
preprocess the dataset by reducing its dimension and then ap-
ply a clustering/classification algorithm. To test whether our
proposed algorithm would preserve clusters in a dataset we pro-
pose the following setting. Assume we want to visualize data
in 3D such that its given 2D projections contain clusters, e.g.,
see the first row of Figure 6. Each subfigure of the first row
of Figure 6 contains 2 clusters, however there is no correspon-
dence between the points, that is, if 2 points are in the same
cluster in one of the subfigures, their position in the other ones
are arbitrary.
Our goal is to apply Algorithm 1 and 2 and see whether the
results preserve the clusters.
The second row of Figure 6 demonstrates the results of the
application of Algorithm 2 to the cluster dataset. We note that
the cluster information is well preserved in all three projections.
We have also applied Algorithm 1 with both projection sets
from (4) and (5). However, Algorithm 2 achieved the best results
3.4 Grid-Path graph Dataset Description
The previous three examples were data/shape visualisation ex-
amples. However, MDS is also used for graph visualization
problems. That is, if one has a weighted/unweighted graph
and wants to visualize it in 2D or 3D such that the vertices
with smaller weights are put close to each other and vertices
with larger weights are drawn far from each other, we can use
MDS on the adjacency matrix. Note that if two vertices are
not connected one can use all pairs shortest path between these
vertices to define distance.
To demonstrate such an example we create two graphs with
100 vertices: one grid and one path; see the first row of Fig-
ure 7. The goal is to apply our proposed algorithm and see
what would be the 3D visualization of such an input such that
one projection demonstrates the first graph and the second one
demonstrates the second graph.
The second row of Figure 7 demonstrates the application of
Algorithm 1 with 2 fixed projections $E_1$ and $E_2$ from (4). Note
that the the algorithm recovers both grid (the left subfigure)
and the path (the right subfigure).
The third row of Figure 7 demonstrates the results of the
application of Algorithm 2 for the dataset described above. In
this case again, the algorithm recovers the grid very well (the
left subfigure) and the path (the right subfigure) is more clear
than the one for fixed projections.
4 Mathematical Background for Algorithms 1 and 2
In this section, we present and derive some of the formulas
for the various gradient functions that are used in our MPSE
algorithms. Our purpose is to assist the reader who wishes
to implement these algorithms. Computation of the gradient
of the multiview-MDS stress function (2) with respect to the
positions $X$ or orthogonal matrices $Q_i$ is cumbersome if the
right matrix calculus tools are not used. We begin with an
overview of the matrix calculus tools that we need and then
derive the gradient functions used in our implementations.
The results are derived for general embedding dimension $p$,
projection rank $r$, and number of projections $L$. However, for
the sake of presentation of the results we limited the exposi-
We begin with an overview of some matrix calculus tools that we use in our derivations of the gradient functions of the multiview-MDS stress function (2).

Let \( y = f(X) \), where \( f : \mathbb{R}^{n \times p} \to \mathbb{R} \). The derivative of \( y \) with respect to \( X \) is the \( p \times n \) matrix given by

\[
\frac{dy}{dX} := \left[ \frac{\partial f}{\partial X_{ij}} \right]_{ij},
\]

where \( X_{ij} \) is the \((i,j)\) entry of \( X \). The gradient of \( y \) with respect to \( X \) is the \( n \times p \) matrix

\[
\nabla_X y := \left[ \frac{\partial f}{\partial X_{ij}} \right] = \left( \frac{dy}{dX} \right)^T.
\]

We use derivative notation to derive the results that we need before changing to gradient notation.

The linearization of \( f \) about \( X_0 \) is given by

\[
y(X_0 + \Delta X) = y(X_0) + \sum_{i,j} \frac{\partial f}{\partial X_{ij}} (\Delta X)_{ij} + \cdots
\]

(to save space, we do not explicitly write the term \( O(\|\Delta X\|_F^2) \), which in terms of differentials takes the form

\[
dy = \text{tr} \left( \frac{dy}{dX} (X_0) dX \right).
\]

We can use properties of differentials and the trace function to simplify the derivation of the various gradient functions.

If \( g : \mathbb{R} \to \mathbb{R} \) and \( z = g(f(X)) \), the chain rule says that

\[
\frac{dz}{dX} = g'(f(X)) \frac{df}{dX} (X).
\]

If \( \Pi \) is a fixed \( p \times p \) matrix and \( y(X) = f(X\Pi^T) \), we have

\[
y(X + \Delta X) = f((X + \Delta X)\Pi^T) = f(X\Pi^T + \Delta X\Pi^T) = f(X\Pi^T) + \text{tr}(\Pi^T \frac{df}{dX} (X)\Delta X) + \cdots,
\]

so that

\[
\frac{d}{dX} \left( f(X\Pi^T) \right) = \Pi^T \frac{df}{dX} (X\Pi^T)
\]

and

\[
\nabla_X \left( f(X\Pi^T) \right) = \nabla f(X\Pi^T) \Pi.
\]

Similarly, if we set \( \Pi = QQ^T \) and differentiate \( y(Q) = f(XQQ^T) \) with respect to \( Q \), it can be shown that

\[
\nabla_Q \left( f(XQQ^T) \right) = \left( \nabla f(XQQ^T) \right)^T X + X^T \nabla f(XQQ^T) Q.
\]

We will use the two previous results in deriving formulas for the \( X \) and \( Q \) gradients of (6).

4.2 Computation of Relevant Gradients

We begin by computing the \( X \) and \( Q \) gradients of the components \( \|
\Pi_x r_i - \Pi_x r_j \| \) of the multiview-MDS stress function (6). The formulas for the \( X \) and \( Q \) gradients of (6) follow easily after that.

Let \( d_{ij}(X) \) be the distance between positions \( r_i \) and \( r_j \). This can be written as

\[
d_{ij}(X) = \|X^T e_i - X^T e_j\|_2 = \|X^T (e_i - e_j)\|_2,
\]

where \( e_i \) and \( e_j \) are the \( i \)-th and \( j \)-th standard basis (column) vectors of \( \mathbb{R}^n \). The square distance can be written as

\[
d_{ij}^2(X) = \|X^T (e_i - e_j)\|^2 = \text{tr} \left( X^T (e_i - e_j)(e_i - e_j)^T X \right),
\]

where

\[
A_{ij} := (e_i - e_j)(e_i - e_j)^T.
\]

Note that

\[
d\text{tr} \left( X^T A_{ij} X \right) = \text{tr} \left( d \left( X^T A_{ij} X \right) \right) = \text{tr} \left( dX^T A_{ij} X + A_{ij} dX \right) = \text{tr} \left( X^T A_{ij} dX \right) = \text{tr} \left( (2X^T A_{ij}) dX \right).
\]
Fig. 6: This first row of this figure presents the original clusters, each subfigure contains the same number of points in 2D with 2 well distinguishable clusters. The second row presents the results of the application of Algorithm 2 for this dataset.

and so
\[
\frac{dd_{ij}^2}{dX}(X) = 2X^T A_{ij}.
\]

It then follows that
\[
\frac{dd_{ij}}{dX}(X) = \frac{d}{dX} \sqrt{d_{ij}^2(X)} = \frac{1}{2\sqrt{d_{ij}^2(X)}} \frac{dd_{ij}^2}{dX}(X),
\]

and that
\[
\nabla d_{ij}(X) = \frac{1}{d_{ij}(X)} A_{ij} X.
\]

If Π is a p × p matrix, we have
\[
\nabla \Xi (d_{ij}(X \Pi^T)) = \nabla_X d_{ij}(X \Pi^T) \Pi = \left( \frac{1}{d_{ij}(X \Pi^T)} A_{ij} (X \Pi^T) \right) \Pi.
\]

If we set Π = QQ^T and differentiate with respect to Q, we similarly obtain
\[
\nabla_Q \left( d_{ij}(X QQ^T) \right) = \frac{1}{d_{ij}(X QQ^T)} \left( QQ^T X^T A_{ij} X + X^T A_{ij} X QQ^T \right) Q.
\]

For a fixed n × n dissimilarity/distance matrix D, the MDS stress function can be written as
\[
S(X; D) = \sum_{i<j} (d_{ij}(X) - D_{ij})^2.
\]

which has the gradient
\[
\nabla S(X; D) = 2 \sum_{i<j} (d_{ij}(X) - D_{ij}) \nabla_X d_{ij}(X),
\]

\[
:= B(X; D) X
\]
distance matrices, then the multiview-MDS stress function is
\[ X \]

The gradient of the multiview MDS stress function with respect to \( X \) is
\[ \nabla_X (S(X^T; D)) = \nabla S(X^T; D) \Pi \]

If \( \Pi \) is a \( p \times p \) (orthogonal projection) matrix, then the matrix \( X^T \Pi \) is the \( n \times p \) matrix whose rows are equal to \( \Pi x_i \). The gradient of the multiview MDS stress function with respect to \( X \) is
\[ \nabla_X (S(X^T; D)) = \nabla S(X^T; D) \Pi \]

If \( \{(\Pi_k, d_k)\}_{k=1}^K \) are \( K \) pairs of \( p \times p \) transformations and \( n \times n \) distance matrices, then the multiview-MDS stress function is
\[ S_M \left( X; \{(\Pi_k, D_k)\}_{k=1}^L \right) := \sum_{k=1}^L S(X^T; D_k) = \sum_{k=1}^L \sum_{i<j} (d_{kj} - (D_k)_{ij})^2 \]

and its gradient is
\[ \nabla_X S_M \left( X; \{(\Pi_k, D_k)\}_{k=1}^L \right) = \sum_{k=1}^L \nabla_X S(X^T; D_k) = \sum_{k=1}^L B(X^T; D_k) X^T \Pi_k \]

This is the expression for the gradient function that we use in Algorithm 1 and as part of Algorithm 2.

The gradient of the MDS stress function with respect to \( Q \) can be computed similarly,
\[ \nabla_Q S(XQQ^T; D) = \left( QQ^T X^T \left( B(XQQ^T; D) \right)^T X + X^T B(XQQ^T; D) XQQ^T \right) Q \]

This is the expression for the gradient function that we use in Algorithm 3.

5 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 as well as multivariate networks. Our initial implementation relies on standard gradient descent, which is expensive given the additional overhead of simultaneous optimizations. We plan to implement a stochastic gradient descent version as well as consider multiview stress majorization.

Acknowledgement
We would like to thank the participants in the "Visualizing Dynamic Graphs in VR" working group at Shonan Seminar 131. Discussion about visualizing dynamic graphs in 3D inspired the (simpler) idea of visualizing static data from different points of view.

This research is supported by National Science Foundation TRIPODS program, grant CCF-1740858.
[19] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inform. Process. Lett.*, 31:7–15, 1989.

[20] Kanaan Izquierdo, A. Ziyatdinov, M. A. Burgueño, and A. Perera Lluna. Multiview: a software package for multiview pattern recognition methods. *Bioinformatics*, (bty1039):1–3, 2018.

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

[22] 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.

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

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

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

[26] 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.

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

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

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

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

[31] 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.

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

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

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

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

[36] 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.

[37] 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.

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