Denoising and completion of Euclidean distance matrix from multiple observations

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Electronics and Communication Engineering by Research

by

Sai Sumanth Natva
2019702018
sai.sumanth@research.iiit.ac.in

Advisor: Dr. Santosh Nannuru

International Institute of Information Technology Hyderabad
500 032, India

June 2024

International Institute of Information Technology Hyderabad
Hyderabad, India

CERTIFICATE

This is to certify that work presented in this thesis proposal titled ***Denoising and completion of Euclidean distance matrix from multiple observations*** by *Sai Sumanth Natva* has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Advisor: Dr. Santosh Nannuru

To everyone

Acknowledgments

I would like to express my deepest gratitude to Dr. Santosh Nannuru for his invaluable guidance, unwavering support, and constant encouragement throughout this journey. Dr. Nannuru's expertise, insights, and dedication have played a pivotal role in shaping not only my research but also my academic endeavors as a whole. His mentorship has been instrumental in helping me navigate through challenges and achieve goals.

Furthermore, I am profoundly thankful to my family for their boundless love, understanding, and unwavering encouragement. Their steadfast belief in my abilities has served as a constant source of strength and motivation, fueling my determination to excel in my academic pursuits. Their unwavering support has been a cornerstone of my journey, inspiring me to persevere through both triumphs and tribulations.

Additionally, I extend my heartfelt appreciation to my friends and labmates for their continuous encouragement and unwavering support. The camaraderie we share, coupled with meaningful discussions and shared experiences, has enriched my academic journey in profound ways, making it not only academically rewarding but also personally fulfilling. Their companionship has created a supportive environment where we inspire and uplift each other, fostering growth and mutual success.

Abstract

For a point set, Euclidean distance matrix (EDM) is the matrix consisting of squared distances between every pair of points in the set. It frequently appears in wide ranging applications such as sensor networks, acoustic arrays, crystallography, and self localization among others. Often the measured EDM are inaccurate (due to noise) and incomplete (due to limited availability of measurements) requiring denoising and completion algorithms, frequently both. We focus on two methods from the literature: the first one is the Rank Alternation algorithm, which is based on the rank property of EDM, and the second one is the Semi-definite Relaxation (SDR) algorithm, which hinges on the mapping between EDMs and Gram matrices. Both have demonstrated success, albeit confined to handling single noisy and/or incomplete input EDM.

In this thesis we propose extensions of these two algorithms which can process multiple noisy and incomplete EDM simultaneously. We verified this through simulations where we varied multiple parameters such as the number of points, the number of EDM inputs, Signal-to-Noise Ratio (SNR) value, amount of missing entries, and noise types such as zero-mean Uniform noise, zero-mean Gaussian noise, and zero-mean Laplacian noise. Simulations show that the proposed joint algorithms (which process multiple EDM) outperform the corresponding individual algorithms (which process individual EDM and combine the results).

Contents

List of Figures

List of Tables

List of Related Publications

[P1] Sai Sumanth Natva, and Santosh Nannuru, **"Denoising and completion of Euclidean distance matrix from multiple observations"**, at *National Conference on Communications*, IIT Madras, Feb 2024.

Chapter 1

Introduction

Imagine we have a set of points in space, but we don't know the exact locations of each point. However, we do have information about the distances between these points. Note that Euclidean distance stands out as the predominant metric for quantifying the spatial separation between two points. For example, in the sensor network localization problem, the goal is to determine the positions of sensor nodes within a network. However, the exact locations of these sensor nodes are initially unknown. Instead, we have access to information about the pairwise distances between the sensor nodes. We can access these pairwise distances using time of arrival (ToA) measurements [1] between sensor pairs. By measuring the time it takes for a signal to travel from one sensor node to another, and knowing the speed of signal propagation, we can calculate the distance between the two nodes.

These distance measurements can further be used to recover the point set configuration [2–5] provided that the location of a few anchor points are known. This can be done using the Orthogonal Procrustes algorithm which is a method used in data analysis to align two sets of points in a way that minimizes the differences between them. Imagine you have two similar shapes on a graph, but one is slightly rotated, translated, or scaled compared to the other. The goal of the Orthogonal Procrustes algorithm is to find the best way to adjust one shape so it matches the other as closely as possible. This algorithm is widely used in fields like computer vision, psychology, and bioinformatics, where it's important to compare shapes or patterns. For example, in face recognition, it might be used to align a new face with a database of known faces to see if there's a match.

Apart from sensor networks [1, 6], EDMs show up in many different applications. One significant application is in molecular shape reconstruction [7], where the primary goal is to identify the three-dimensional structure of a molecule using prescribed distances between atoms. This is crucial in understanding molecular interactions and functions. In this problem, interaction data between atoms within a molecule is expressed as a series of Euclidean distances, organized into an EDM. A notable method used in this domain involves nuclear magnetic resonance (NMR) spectroscopy [8]. NMR provides distance constraints between atoms in a molecule, which can be converted into an EDM. Advanced distance geometry programs are employed to compute the complete spatial structures of molecules, particularly small proteins, from NMR [9]. For example, geometric constraints derived from the crystal structure of the basic pancreatic trypsin inhibitor have been used to generate conformers that match

NMR-derived distance constraints. These computed structures are then compared with each other and the original crystal structure, demonstrating the effectiveness of NMR in determining the global conformation of polypeptide chains. Moreover, this comparison provides a benchmark for evaluating the quality of protein structures computed from NMR data when no crystal structure is available, guiding improvements in NMR experiments for protein structure determination. Overall, EDMs showcases a powerful approach in structural biology, enhancing our understanding of molecular conformations and interactions.

In the field of psychometrics [10], multidimensional scaling (MDS) involves several processes such as determining the comparative distances between stimuli, estimating unknown constants, and assessing the dimensionality of psychological space. One key aspect of MDS is the transformation of these comparative distances into absolute distances, which is often achieved using techniques like least-squares solutions and Euclidean models. EDMs are particularly useful in this context. They help in representing the distances between different stimuli in a way that can be visualized and analyzed. By using EDMs, researchers can project stimuli onto axes within a psychological space, making it easier to interpret and understand the relationships and differences between them. This projection allows for a clearer visualization of how different stimuli are perceived relative to each other, which is crucial for various applications in psychology and related fields.

In the realm of machine learning, manifolds are essential for representing high-dimensional data in lower-dimensional spaces, making the data easier to understand and analyze. EDMs are a key tool in this process, as they help to reduce the dimensionality while preserving the local geometry of the data. This means that the relationships between data points are maintained, even in the lower-dimensional representation. For instance, EDMs have been used to learn image manifolds [11], which allows for effective image processing and recognition. By representing images in a reduced dimension, it becomes easier to identify patterns and similarities. Similarly, in handwriting recognition [12], EDMs help in mapping handwritten characters into a lower-dimensional space where their distinctive features can be more easily recognized and analyzed. These applications, along with many others [13], demonstrate how EDMs contribute to various machine learning tasks by simplifying complex, high-dimensional data while retaining important structural information.

In acoustics, EDMs are crucial for determining the structure of a convex polyhedral room, as described in a study by Dokmanic et al [14]. Hearing the shape of a room using a finger snap, while being blindfolded, is a difficult task. It is challenging to design computer algorithms for the same purpose. To tackle this, the authors utilized sound recordings from a set of microphones placed inside the room, capturing the echoes generated by a finger snap. By analyzing the geometric relationships among the arrival times of these echoes, the room's geometry can be estimated using the properties of EDMs. The study further shows that, under certain conditions, first-order echoes alone can uniquely describe convex polyhedral rooms. The algorithm begins with the recorded impulse responses and learns to accurately assign echoes to the corresponding walls. Also here, we don't have any access to the information about microphone array's configuration. As long as the microphones can capture the echoes, their placement

within the room is flexible. In the emerging field of Internet of Things (IoT), it is useful for studying network localization [15]. It has been the subject of at least one doctoral dissertation [16].

Processing Euclidean Distance Matrices (EDMs) presents significant challenges due to the inherent noise and incompleteness in the data. This noise and missing information can lead to inaccuracies in the results, complicating the analysis. In the realm of EDM analysis, researchers have devised various methodologies to tackle these issues effectively. One prominent example is the development of Rank Alternation and Semi-Definite Relaxation (SDR) methods, as discussed in [13]. These methods are particularly useful in scenarios where a single EDM is provided as input. Rank Alternation helps in refining the rank of the matrix iteratively to better approximate the true distances, while SDR methods relax the problem into a semi-definite programming framework, which is easier to solve and provides a robust solution even in the presence of noise and missing data. These advancements significantly enhance the accuracy and reliability of EDM analysis, making them crucial tools in the field.

These algorithms are primarily designed to process a single noisy and incomplete instance of an EDM. However, in real-world situations, we can have access to multiple measurements of the EDM, each potentially containing its own noise and incompleteness . The difficulty lies in integrating these multiple observations into a unified EDM estimate, as these algorithms are not originally built to handle multiple inputs simultaneously. This makes it challenging to effectively merge the data from each measurement. To overcome this limitation, we suggest enhancing these algorithms to allow for the simultaneous processing of multiple EDM observations. By doing so, we can utilize the information from each observation to create a more accurate and robust estimate of the true underlying EDM.

The study detailed in [17] explores the application of Kinetic Euclidean Distance Matrices (KEDM) to analyze object movements along trajectories, with a primary focus on reconstructing these trajectories from incomplete and noisy distance observations taken at multiple time instances. This involves using EDMs to represent the pairwise distances between moving objects at various points in time and developing optimization techniques to accurately reconstruct the trajectories despite the challenges posed by missing data and measurement noise. Furthermore, [18] focus on constructing coherent and precise representations of similarity from multiple incomplete and noisy approximations.

Our work can be considered a special case of KEDM where the objects remain stationary, but distance measurements are taken at multiple instances over time. In this scenario, it is as if we have multiple versions of the true underlying EDM, each reflecting the positions of the objects. However, these measurements are subject to noise and incompleteness due to various factors, such as sensor inaccuracies, environmental interference, or data loss. This introduces challenges in accurately capturing the true distances between objects, necessitating robust methods to handle these imperfections in the data. In many such scenarios, it is possible that multiple EDM observations are available (for example, sensors communicate more than once in a network).

The thesis is organized as follows: brief review of EDM and existing algorithms is given in Chapter 2; novel algorithms which can process multiple EDM observations are proposed in Chapter 3 along with the performance comparison of these algorithms, and Chapter 4 concludes the thesis.

In this thesis, we represent scalars by lowercase letters, vectors by lowercase bold letters, and matrices by uppercase bold letters. $\|.\|_F$ denotes the Frobenius norm of a matrix, and $\circ$ stands for the Hadamard (entry wise) product between matrices [13]. $\mathbf{1}$ represents the column vector of all ones and diag() represents diagonal of a matrix using a column vector.

Chapter 2

Background review

This chapter delves into EDMs and offers a concise review of earlier methods suggested for denoising and completing EDM observations.

2.1 Construction of EDMs

Let the $d \times n$ matrix $\mathbf{X}$ denote the coordinates of n points in d-dimensional space. The Euclidean distance ($\mathbf{d}_{ij}$) between two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in d-dimensional space is calculated as:

$$\mathbf{d}_{ij} = \sqrt{\sum_{k=1}^{d} (x_{ki} - x_{kj})^2} \tag{2.1}$$

$$= \sqrt{\sum_{k=1}^{d} x_{ki}^2 - 2x_{ki}x_{kj} + x_{kj}^2} \tag{2.2}$$

The EDM is formed by the squared distances between each pair of points in a point set. Thus the EDM for point set $\mathbf{X}$ can be written compactly as

$$\mathbf{D} = \mathbf{1}\text{diag}(\mathbf{X}^T\mathbf{X})^T - 2\mathbf{X}^T\mathbf{X} + \text{diag}(\mathbf{X}^T\mathbf{X})\mathbf{1}^T \tag{2.3}$$

where $\mathbf{1}$ represents the column vector of all ones. We can also represent EDMs using the Gram matrix ($\mathbf{G}$) [19]. The Gram matrix, also referred to as the Gramian matrix, is constructed using inner products between a set of vectors. In our case, these vectors correspond to points in a point set. Thus, using the Gram matrix $\mathbf{G} = \mathbf{X}^T\mathbf{X}$, the EDM can be expressed as follows.

$$\mathbf{D} = \mathcal{K}(\mathbf{G}) = \mathbf{1}\text{diag}(\mathbf{G})^T - 2\mathbf{G} + \text{diag}(\mathbf{G})\mathbf{1}^T. \tag{2.4}$$

These diverse representations of EDMs offer insights into various properties of EDMs. Understanding these properties facilitates the development of tailored algorithms for denoising and completion tasks on EDMs.

2.2 Estimation of Point Sets from EDMs

When we construct EDM from point set, it's essential to carefully monitor what information we can retain and what we might lose in the inverse problem. In our specific case, the dimensionality of the point set is $d \times n$, while the resulting EDM is $n \times n$. Typically, in practical scenarios, the number of points far exceeds the dimensionality of the point set. This means that when we convert a point set into an EDM, we're effectively increasing the number of parameters required to represent the point set.

Additional information beyond what's contained in the EDMs is necessary to recreate the point set accurately. This lost information often includes details such as the absolute position of the points in space, and their orientations. Therefore, algorithms attempting to reconstruct the original point set solely from EDM can only produce a solution that matches the original configuration up to a rigid transformation, such as translation, rotation, or reflection. Different point sets can be obtained for different choices of coordinate systems. One such process is discussed in [20]. This process is known as classical Multidimensional Scaling (MDS). This limitation highlights the importance of additional information beyond EDM for accurately reconstructing the original point set.

The Orthogonal Procrustes algorithm is designed precisely for this purpose [21]. It operates with two key inputs. The first one is the anchor point set ($\mathbf{Y}$) ; i.e., we are given the original locations of a few points in the point set. The second one is the reconstructed or estimated point matrix corresponding to anchor points ($\mathbf{X_a}$); i.e., after estimating the entire point set ($\mathbf{X}$) up to a rigid transformation, we only pick the points corresponding to anchor points in $\mathbf{Y}$ from it. The aim of this algorithm is to best fit the subset of the estimated point set considered ($\mathbf{X_a}$) to the anchor point set ($\mathbf{Y}$). This alignment process may involve rotations, reflections, and translations to minimize differences between the matrices $\mathbf{X_a}$ and $\mathbf{Y}$.

The Procrustes analysis involved in this algorithm consists of the following steps. In the first step, the geometric centers of $\mathbf{X_a}$ and $\mathbf{Y}$ are computed as $\mathbf{x_{a,c}}$ and $\mathbf{y_c}$, respectively. The geometric center, also known as the centroid, of a set of points is the average position of all the points in the set. Given a point set $\mathbf{X}$ with n points in d-dimensional space, represented as a $d \times n$ matrix where each column represents a point, the geometric center (or centroid) $\mathbf{x_c}$ is computed as follows:

$$\mathbf{x_c} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \tag{2.5}$$

where, $\mathbf{x}_i$ represents the i-th point in the set $\mathbf{X}$.

Then, both point sets are moved by their respective geometric centers (represented as $\overline{\mathbf{X}}_a$ and $\overline{\mathbf{Y}}$), which makes their centroids coincide. This step accounts for translation between both point sets. In the

next step, we need to find the rotation/reflection that needs to be applied on $\overline{\mathbf{X}}_a$ to align it in the best possible way to $\overline{\mathbf{Y}}$. Note that the rotation/reflection matrix $\mathbf{R}$ here is an orthogonal matrix following the orthogonality property. Thus, $\mathbf{R}$ can be computed as follows.

$$\mathbf{R} = \underset{\mathbf{Q}:\mathbf{Q}\mathbf{Q}^\top=\mathbf{I}}{\arg\min} \|\mathbf{Q}\overline{\mathbf{X}}_a - \overline{\mathbf{Y}}\|_F^2. \tag{2.6}$$

Schönemann in his PhD thesis [22], mentioned a solution to the Equation (2.6). it is obtained using Singular Value Decomposition (SVD) of the matrix $\overline{\mathbf{X}}_a\overline{\mathbf{Y}}^\top$. Let's assume that the SVD of $\overline{\mathbf{X}}_a\overline{\mathbf{Y}}^\top$ is $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$. Then the solution proposed is, $\mathbf{R} = \mathbf{V}\mathbf{U}^\top$. Thus, now from the estimated point set and the anchor point set, we can obtain the original point set through Orthogonal Procrustes algorithm ($\mathbf{X}_{or}$) as follows :

$$\mathbf{X}_{or} = \mathbf{R}(\mathbf{X} - \mathbf{x}_{a,c}\mathbf{1}^\top) + \mathbf{y}_c\mathbf{1}^\top. \tag{2.7}$$

2.3 Noisy and Incomplete EDMs

Usually, in very rare cases, we deal with accurate EDMs corresponding to a point set. In all other cases, we have EDMs with some defects, such as noise, incompleteness, or sometimes both. This is quite common due to various reasons. Some such cases are discussed next.

2.3.1 Cases of Noisy EDMs

Here are a few scenarios where EDM entries are corrupted by noise.

1. Measurement Errors : While measuring distances between points, errors may arise from equipment limitations or human oversight [23], which in turn introduce inaccuracies into the EDM, causing noise in its entries.

2. Sensor Noise : In scenarios where sensor networks or sensor-collected data are used, noise stemming from sensor inaccuracies or interference can distort distance measurements [24], directly impacting the accuracy of the EDM entries.

3. Quantization Errors : Within digital systems, distance measurements frequently undergo quantization due to limitations in resolution or precision. Quantization errors [25] emerge as continuous distance values are rounded to discrete levels, thereby introducing noise into the EDM.

4. Imputation Methods : In certain instances, incomplete entries in the EDM can be approximated or filled using interpolation, extrapolation, or alternative data imputation methods [26]. Nonetheless, employing these techniques may introduce further uncertainty or noise into the EDM, especially when the missing data is inadequately constrained. For instance, within geographical mapping applications, estimating missing distance measurements between locations might rely on geographic features or road

networks. However, the accuracy of such estimates is contingent upon the quality of available geographic data and the complexity of the terrain.

5. Constrained Distance Measurements : Sometimes, the distance measurements in EDMs are not accurately given but are restricted to a range [27]. In this case, the distance measurements in the EDM deviate from the original measurements, containing noise.

6. Environmental Factors : In practical scenarios such as robotics or autonomous vehicles, environmental factors like terrain variations, or atmospheric disturbances [28] can impact distance measurements, resulting in noisy entries in the EDM.

Various applications in which EDMs are noisy will mostly fall under one of these categories.

2.3.2 Cases of Incomplete EDMs

Encountering situations where certain entries are missing is a common occurrence when working with EDMs. Let's explore some scenarios where this happens and the challenges it poses:

1. Limitations in Data Collection: In practical scenarios, attaining full pairwise distance measurements between all points within a pointset may not be possible consistently. This constraint can stem from various factors, including limitations in data collection methodologies, resource constraints, or technical impediments. As an illustration, within an environmental monitoring sensor network, specific sensors could experience connectivity problems or sensor malfunctions [29] [30]. Consequently, this may lead to gaps in the EDM since these sensors are unable to capture distance measurements.

2. Sparse Data : In extensive datasets or spaces with high dimensions, acquiring complete pairwise distance measurements for all points can be both computationally demanding and impractical. Consequently, only a subset of pairwise distances may be computed or accessible, resulting in incomplete entries within the EDM. This scenario is notably prevalent in fields like molecular modeling [27], where calculating pairwise distances between atoms or molecules within intricate systems can require significant computational resources.

3. Noise and Inaccuracies : When noise or inaccuracies impact distance measurements, certain entries in the EDM may become unreliable and thus should be excluded. For example, in computer vision applications, distance measurements between image keypoints could be influenced by factors such as image noise, occlusions, or inaccuracies in feature detection algorithms [31]. As a result, this could result in the absence of unreliable entries in the EDM.

4. Sparse Data : When only a subset of pairwise distances between points is accessible, the EDM might lack completeness. Attempting to estimate or infer missing entries is common, yet this approach can still introduce noise, especially if the estimation of missing data is not precise [32].

Effectively managing the issue of missing entries in EDMs necessitates a thorough examination of the data generation process. Methods for addressing missing data in EDMs encompass various strategies, including data imputation techniques, resilient distance metrics capable of handling missing values, and algorithmic approaches designed to handle data sparsity.

2.3.3 Representation of Noisy and Incomplete EDMs

As discussed earlier, EDMs, in practical terms, are noisy and incomplete. Let's consider an EDM constructed from n points. Then there are $(1/2)n(n-1)$ number of pairs of points present in total. Let's say that the distance information is present only between $m \leqslant (1/2)n(n-1)$ pairs of points belonging to set $\mathbf{S}$. The information regarding these m pairs of points is represented in the EDM by filling the known distances at the places corresponding to these pairs. So the EDM has $2m$ entries filled in it (as it is symmetric in nature), representing this information. The rest of the entries in the EDM indicate that there is no information known between those corresponding pairs of points. Also, in general, the EDM entries are noisy in nature. Thus, every known entry in the EDM or every entry corresponding to the pairs in $\mathbf{S}$ is represented as follows.

$$\tilde{\mathbf{d}}_{ij} = \mathbf{d}_{ij} + \boldsymbol{\varepsilon}_{ij}, \tag{2.8}$$

where $\boldsymbol{\varepsilon}_{ij}$ is the noise corrupting the distance information $\mathbf{d}_{ij}$ between the pair (i, j).

To represent an incomplete EDM, we use a matrix called as Mask matrix ($\mathbf{W}$), which indicates which information is present and which is not in the EDM. The entries in the mask matrix are as follows:

$$\mathbf{w}_{ij} \overset{\text{def}}{=} \begin{cases} 1, (i,j) \in \mathbf{S} \\ 0, \text{otherwise.} \end{cases} \tag{2.9}$$

Thus, we can only operate on the known elements of the EDM by using the entrywise product (Hadamard poduct ($\circ$)) with the mask matrix $\mathbf{W}$. Let's consider a case where we have two matrices $\mathbf{A}$ and $\mathbf{B}$, both having a few missing entries in identical locations. To compare how close $\mathbf{A}$ and $\mathbf{B}$ are to each other, we compute the Frobenius norm, $\|\mathbf{W} \circ (\mathbf{A} - \mathbf{B})\|_F$. This norm is computed based only on the known entries in both matrices using their mask matrix $\mathbf{W}$.

Additionally, to represent that matrix $\mathbf{A}$ has known elements in the locations corresponding to mask matrix $\mathbf{W}$, the notation $\mathbf{A}_W$ is used. Similarly, to enforce the known entries of matrix $\mathbf{A}$ to the corresponding known locations of matrix $\mathbf{B}$ (both having the same mask matrix), we use the notation $\mathbf{A}_W \leftarrow \mathbf{B}_W$.

2.4 Properties of EDMs

EDMs exhibit several properties that are fundamental to their use and understanding:

2.4.1 Diagonal Elements

The zero values in the diagonal elements of an EDM arise from the property that the distance between a point and itself in Euclidean space is always zero.

$$\mathbf{d}_{ii} = 0$$

2.4.2 Non-negativity

In EDMs, each entry represents the squared distance between a pair of points. Given that squared distances are inherently non-negative (as they involve the square of a real number), it's straightforward to see that all entries in EDMs are non-negative.

$$\mathbf{d}_{ij} \geq 0$$

2.4.3 Symmetry

In Euclidean space, the distance between two points is the same regardless of the order in which they are considered. Thus, the EDM is symmetric, meaning the EDM entry corresponding to distance from point i to point j is same as that of the EDM entry corresponding to distance from point j to point i.

$$\mathbf{d}_{ij} = \mathbf{d}_{ji}$$

2.4.4 Triangle Inequality

This property ensures that the sum of two sides of a triangle is always greater than or equal to the length of the third side. In the context of EDMs, it means that the direct distance between two points is always less than or equal to the sum of the distances via any intermediate point.

$$\sqrt{\mathbf{d}_{ij}} \leqslant \sqrt{\mathbf{d}_{ik}} + \sqrt{\mathbf{d}_{kj}}$$

2.4.5 Scaling Invariance

EDMs maintain their properties when the distances are scaled by any positive factor. This means that if $\mathbf{D}$ is a EDM, then raising each element to a positive power s (where $0 < s < 1$), i.e., $(\mathbf{d}_{ij})^s$, still yields a EDM [33].

2.4.6 Rank

The rank property of the EDM states that the rank of an EDM corresponding to a point set in d-dimensional space is at most $d + 2$. This can be derived from Equation (2.3), which consists of three terms. Given that $\mathbf{X}$ is the point set in d-dimensional space with n points, the rank of $\mathbf{X}$ is at most d (as

$d << n$ in general). This implies that the rank of $\mathbf{X}^\top\mathbf{X}$ is also at most d. The other two terms in the equation, $\mathbf{1}\,\text{diag}(\mathbf{X}^\top\mathbf{X})^\top$ and $\text{diag}(\mathbf{X}^\top\mathbf{X})\mathbf{1}^\top$, are rank-1 matrices.

According to the rank inequality, the rank of the sum of matrices is at most the sum of the ranks of the corresponding matrices. The rank property of the EDM is based on this rank inequality.

One very useful observation we can make from this property is that, regardless of the number of points in the point set, the rank of the EDM will always be at most $d + 2$. In practice, there may be thousands of points in the point set, and the dimension will typically be three or less. In such cases, the rank of the EDM is at most five. Therefore, what matters in this property is the smallest affine dimension of the point set (aff dim($\mathbf{X}$)), encompassing all the points.

2.4.7 Positive semi-definiteness

Consider the geometric centering matrix

$$\mathbf{J} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T, \tag{2.10}$$

which changes the point set such that it is centered at origin, i.e., the set of points given by $\mathbf{XJ}$ has its centroid at origin. Here $\mathbf{I}_n$ denotes the $n \times n$ identity matrix.

The corresponding geometrically centered Gram matrix $\left(\mathbf{G}_c = (\mathbf{XJ})^\top(\mathbf{XJ}) = -(1/2)\mathbf{JDJ}\right)$ of an EDM $\mathbf{D}$, formed by using inner products between the points in a geometrically centered point set $\mathbf{XJ}$, is a positive semi-definite (PSD) matrix [34], [35]. This establishes a mapping between the set of EDMs ($\mathbb{EDM}^n$) and the combination of two sets based on the Gram matrix, namely the set of symmetric positive semi-definite matrices (S^n_+) and the set of geometrically centered matrices ($\mathbb{S}^n_c$). The set of geometrically centered matrices is typically represented as follows.

$$\mathbb{S}^n_c = \{\mathbf{G} \in \mathbb{R}^{n\times n} | \mathbf{G} = \mathbf{G}^\top, \mathbf{G1} = \mathbf{0}\}. \tag{2.11}$$

2.5 EDM denoising and completion algorithms

In practical applications, the raw EDMs that we measure are generally noisy and incomplete. Thus, we need to refine them to reconstruct the point set as accurately as possible. For this purpose, several algorithms have been designed [13]. Among them, we consider two of the most reliable algorithms to perform this task, and they are as follows:

2.5.1 Rank Alternation Algorithm

This algorithm [13], operates by utilizing the rank property to handle a noisy and incomplete input EDM within the targeted embedding dimension of the point set (d). It employs an iterative approach, where it iteratively enforces the known entries of the EDM and exploits the rank property to ensure that the EDM becomes complete while maintaining a rank of at most $d + 2$.

To enforce this rank property, the algorithm initially performs the singular value decomposition (SVD) of the EDM $\mathbf{D}$. The SVD of the matrix $\mathbf{D}$ decomposes it into three matrices: $\mathbf{U}$, $\mathbf{P}$, and $\mathbf{R}$, where $\mathbf{U}$ and $\mathbf{R}$ are $n \times n$ orthogonal matrices, and $\mathbf{P}$ is an $n \times n$ diagonal matrix containing the singular values λ_i on its diagonal. The decomposition can be written as:

$$\mathbf{D} = \mathbf{U}\mathbf{P}\mathbf{R}^T \tag{2.12}$$

The algorithm now retains only the largest $d + 2$ singular values and sets the remaining singular values to zero. This is done to enforce the rank constraint, ensuring that the rank of the resulting matrix is at most $d + 2$. The modified diagonal matrix $\widetilde{\mathbf{P}}$ can be written as:

$$\widetilde{\mathbf{P}} = \text{diag}(\lambda_1, \ldots, \lambda_{d+2}, 0, \ldots, 0) \tag{2.13}$$

Here, $\lambda_1, \ldots, \lambda_{d+2}$ are the largest $d + 2$ singular values, and the remaining entries are set to zero. The corresponding reconstructed EDM $\hat{\mathbf{D}}$ is now formed using the product of matrices $\mathbf{U}$, $\widetilde{\mathbf{P}}$ and $\mathbf{R}$.

$$\hat{\mathbf{D}} = \mathbf{U}\widetilde{\mathbf{P}}\mathbf{R}^T \tag{2.14}$$

The matrix $\hat{\mathbf{D}}$ has a rank of at most $d + 2$ because $\widetilde{\mathbf{P}}$ contains only $d + 2$ non-zero singular values. The algorithm initializes the unobserved entries of the incomplete input EDM with the average of the observed entries, providing a starting point for the iterative process. However, it's worth noting that relying solely on the rank property may not fully characterize the EDM, leading to potential convergence issues. The algorithm is relatively straightforward to implement. Furthermore, it has demonstrated a good success rate in practice [13].

The pseudo code of Rank Algorithm is given in Algorithm 1.

2.5.2 SDR Algorithm

The Semi-Definite Relaxation (SDR) algorithm is based on the mapping between EDMs and Gram matrices that we discussed earlier. By utilizing this mapping, which leverages the positive semidefiniteness property of Gram matrices, we can frame EDM denoising and completion tasks as semidefinite programs [13], [36], [37]. In this mapping, we focus on EDMs corresponding to a dimension d, containing n points. Typically, in practical scenarios, the dimensionality of a point set is much lower compared to the number of points in the set ($d \ll n$). Consequently, these EDMs map to Gram matrices with a rank of d.

As per the positive definiteness property of the Gram matrix, the EDMs corresponding to dimension d are mapped to Gram matrices with the following conditions: Firstly, the geometrically centered Gram matrix ($\mathbf{JDJ}$) corresponding to the EDM should have a rank of d. Secondly, it should be a positive semidefinite matrix. Therefore, by utilizing these constraints, a semidefinite program is formulated as

Algorithm 1 Rank Alternation Algorithm

function `RankAlternation` $(\mathbf{W}, \widetilde{\mathbf{D}}, d)$
  $\mathbf{D}_{\mathbf{W}} \leftarrow \widetilde{\mathbf{D}}_{\mathbf{W}}$: Initialize observed entries
  $\mu \leftarrow$ Average of observed entries
  $\mathbf{D}_{\mathbf{1}\mathbf{1}^T - \mathbf{W}} \leftarrow \mu$: Initialize unobserved entries
  repeat
    $\mathbf{D} \leftarrow \text{RankEnforce}(\mathbf{D}, d+2)$: Enforce rank property
    $\mathbf{D}_{\mathbf{W}} \leftarrow \widetilde{\mathbf{D}}_{\mathbf{W}}$: Enforce known entries
    $\mathbf{D}_{\mathbf{I}} \leftarrow \mathbf{0}$: Set the diagonal to zero
    $\mathbf{D} \leftarrow (\mathbf{D})_+$: Zero the negative entries
  until Convergence or MaxIter
  return $\mathbf{D}$
end function

function `RankEnforce` $(\mathbf{D}, d)$
  $\mathbf{U}, [\lambda_i]_{i=1}^n, \mathbf{R} \leftarrow \text{SVD}(\mathbf{D})$
  $\widetilde{\mathbf{P}} \leftarrow \text{diag}(\lambda_1, \ldots, \lambda_d, \underbrace{0, \ldots, 0}_{n-d \text{ zeroes}})$
  $\hat{\mathbf{D}} \leftarrow \mathbf{U}\widetilde{\mathbf{P}}\mathbf{R}^T$
  return $\hat{\mathbf{D}}$
end function

follows to find the best possible Gram matrix corresponding to input EDM by solving for the problem.

$$\begin{aligned} \underset{\mathbf{G}}{\text{minimize}} \quad & \|\mathbf{W} \circ (\widetilde{\mathbf{D}} - \mathcal{K}(\mathbf{G}))\|_F^2 \\ \text{subject to} \quad & \text{rank}(\mathbf{G}) \leq d \\ & \mathbf{G} \in \mathbb{S}_+^n \cap \mathbb{S}_c^n \end{aligned} . \tag{2.15}$$

where $\widetilde{\mathbf{D}}$ represents the noisy and incomplete EDM input and $\mathbf{W}$ is its corresponding mask matrix. The second constraint in the above problem can be denoted as constraints $\mathbf{G} \succcurlyeq 0$ and $\mathbf{G}\mathbf{1} = \mathbf{0}$. Here, $\mathbf{G} \succcurlyeq 0$ implies that $\mathbf{G}$ is a positive semi-definite matrix, meaning all its eigenvalues are non-negative. The notation $\mathbf{G}\mathbf{1} = \mathbf{0}$ indicates that the sum of each row (or column, since $\mathbf{G}$ is symmetric) of $\mathbf{G}$ is zero, which corresponds to the property of being geometrically centered.

However, we encounter an issue with the first constraint: the rank constraint introduces nonconvexity to the problem [38], making it challenging to find a solution. Thus, to eliminate this rank constraint [39], allowing us to obtain a solution for the Gram matrix whose rank can correspond to an appropriate dimension. Consequently, the semidefinite program now transforms as follows.

$$\begin{aligned} \underset{\mathbf{G}}{\text{minimize}} \quad & \|\mathbf{W} \circ (\widetilde{\mathbf{D}} - \mathcal{K}(\mathbf{G}))\|_F^2 \\ \text{subject to} \quad & \mathbf{G} \in \mathbb{S}_+^n \cap \mathbb{S}_c^n \end{aligned} . \tag{2.16}$$

However, another concern arises with this problem. As the constraint $\mathbf{G1} = \mathbf{0}$ suggests, $\mathbf{G}$ has a null space, indicating that there is at least one eigenvalue that is zero. This implies that a solution for a strictly positive definite Gram matrix cannot be achieved [39]. Therefore, we need to address this issue by removing the particular part responsible for it. Thus, an invertible transformation is employed to utilize a reduced version of the current Gram matrix [13], ensuring that the aforementioned problem does not arise again.

To establish an invertible transformation for the purpose discussed above, we require an orthogonal basis of dimension $(n-1)$, denoted as $\mathbf{V}$. It is an $n \times (n-1)$ matrix with certain constraints. Firstly, being an orthonormal basis, it can be mathematically represented by $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$. Additionally, we address the constraint $\mathbf{G1} = \mathbf{0}$ through the invertible transformation between the original Gram matrix and the reduced Gram matrix, utilizing $\mathbf{V}$ in a manner that satisfies the following equation.

$$\mathbf{V}^\top\mathbf{1} = \mathbf{0} \tag{2.17}$$

Now, the matrix $\mathbf{V}$ is chosen as follows to satisfy all the above constraints.

$$\mathbf{V} = \begin{bmatrix} p & p & \cdots & p \\ 1+q & q & \cdots & q \\ q & 1+q & \cdots & q \\ \vdots & \cdots & \ddots & \vdots \\ q & q & \cdots & 1+q \end{bmatrix}, \tag{2.18}$$

where $p = -1/\sqrt{(}n)$ and $q = -1/(n + \sqrt{(}n))$.

The invertible transformation between the original Gram matrix $\mathbf{G}$ and the reduced Gram matrix $\mathbf{H}$, as discussed above, is represented as follows.

$$\mathbf{G} = \mathbf{V}\mathbf{H}\mathbf{V}^T. \tag{2.19}$$

Hence, we can revise the semi-definite program in (2.16) to the below-mentioned program, which makes use of the reduced Gram matrix to solve the issue and provides better numerical stability compared to (2.16) [36].

$$\begin{aligned} \underset{\mathbf{H}}{\text{minimize}} \quad & \|\mathbf{W} \circ (\tilde{\mathbf{D}} - \mathcal{K}(\mathbf{V}\mathbf{H}\mathbf{V}^\top))\|_F^2 \\ \text{subject to} \quad & \mathbf{H} \in \mathbb{S}_+^{n-1} \end{aligned}. \tag{2.20}$$

The feasible solution set for Equation (2.20) will be the semi-definite matrices of dimension $(n-1)$. However, since we relaxed the rank constraint in the program, the solution will usually correspond to higher dimensionality, though there is a chance that a solution exists corresponding to dimension d. There are many strategies to tackle this issue, out of which adding the trace norm to the program is one effective way to do it.

The addition of the trace norm to the semi-definite program also helps to make it convex. Here, we focus on maximizing the trace norm of $\mathbf{G}$. According to [13], maximizing the trace norm of $\mathbf{G}$ is akin to maximizing the sum of squared distances between the points. The authors explained that this process will stretch the configuration of the point set as much as possible, within the given constraints. They used analogies such as unrolling a piece of paper or straightening a bent string to illustrate that stretching promotes configurations with smaller affine dimensions. It serves as a regularization technique that can improve the semi-definite program (2.20) to obtain a more accurate solution [40]. The authors have successfully applied this concept in manifold learning, while Biswas et al. utilized it in Sensor network localization (SNL) [37].

Since the traces of both matrices $\mathbf{H}$ and $\mathbf{G}$ are equal, the improved semi-definite program is as follows:

$$\begin{aligned} \underset{\mathbf{H}}{\text{maximize}} \quad & \text{trace}(\mathbf{H}) - \lambda \|\mathbf{W} \circ (\widetilde{\mathbf{D}} - \mathcal{K}(\mathbf{V}\mathbf{H}\mathbf{V}^T))\|_F \\ \text{subject to} \quad & \mathbf{H} \in \mathbb{S}_+^{n-1}, \end{aligned} \tag{2.21}$$

The choice of λ in Equation (2.21) is set to 100. We determined this value after conducting numerous experiments to find the optimal λ, testing values in the range of $[0.1, 1000]$. Solving this optimization problem, we obtain the solution for the reduced Gram matrix $\mathbf{H}$. Then, using the invertible transformation (2.19), we can construct the original Gram matrix $\mathbf{G}$ from the obtained reduced Gram matrix $\mathbf{H}$. The EDM can be constructed using Equation (2.4) from the Gram matrix $\mathbf{G}$. The CVX implementation of the SDR algorithm [41], [42] is shown in Algorithm 2.

Algorithm 2 SDR Algorithm

function SDR ($\mathbf{D}, \mathbf{W}, \lambda$)
- $n \leftarrow \text{size}(\mathbf{D}, 1)$
- $x \leftarrow -1/\left(n + \text{sqrt}(n)\right)$
- $y \leftarrow -1/\text{sqrt}(n)$
- $\mathbf{V} \leftarrow [y * \text{ones}(1, n-1);\ x * \text{ones}(n-1) + \text{eye}(n-1)]$
- $\mathbf{e} \leftarrow \text{ones}(n, 1)$
- **cvx_begin sdp** : Begin of CVX implementation
  - variable $\mathbf{H}(n-1, n-1)$ symmetric : Defining reduced Gram matrix
  - $\mathbf{G} \leftarrow \mathbf{V}\mathbf{H}\mathbf{V}^T$
  - $\mathbf{C} \leftarrow \text{diag}(\mathbf{G}) * \mathbf{e}' + \mathbf{e} * \text{diag}(\mathbf{G})' - 2 * \mathbf{G}$
  - maximize $\text{trace}(\mathbf{H}) - \lambda * \text{norm}\left(\mathbf{W} \circ (\mathbf{C} - \mathbf{D}), \text{'fro'}\right)$: Defining Optimization problem
  - **subject to**
    - $\mathbf{H} \geq 0$
- **cvx_end** : End of CVX implementation
- $\mathbf{G} \leftarrow \mathbf{V}\mathbf{H}\mathbf{V}^T$: Constructing Gram matrix
- $\mathbf{D} \leftarrow \text{diag}(\mathbf{G}) * \mathbf{e}' + \mathbf{e} * \text{diag}(\mathbf{G})' - 2 * \mathbf{G}$: Constructing EDM
- **return** $\mathbf{D}$

end function

2.5.3 Other Algorithms for EDM Denoising and Completion

The Alternating Descent algorithm [13] is another well-known method for processing a single EDM input. It focuses on updating a single coordinate of one point at a time, utilizing the separability of the below-mentioned s-stress function (A widely recognized cost function discussed in [43]) across points i and coordinates j. This s-stress function for the i^{th} point and j^{th} coordinate is a fourth-order polynomial and is as follows:

$$f(x;\boldsymbol{\beta}^{(i,j)}) = \sum_{\ell=0}^{4} \beta_{\ell}^{(i,j)} x^{\ell} \tag{2.22}$$

where $\boldsymbol{\beta}$ represents the coefficients of the polynomial for the i^{th} point and j^{th} coordinate. Expressions for these coefficients are given in [16]. The global minimizer of the s-stress function for the i^{th} point and j^{th} coordinate can be obtained by its derivative, which is a cubic polynomial. This approach is then repeated for all the coordinates across all the points. This algorithm uses distributed implementation and is simple to implement.

2.6 Drawbacks of Existing Algorithms

The algorithms described in the literature are typically designed to handle a single noisy and incomplete observation of EDM. However, in practical scenarios, we might have access to multiple measurements of the EDM, each potentially containing noise and incompleteness, but all stemming from the same underlying true EDM.

The challenge arises when we attempt to combine these multiple observations into a single coherent EDM estimate using existing algorithms. These algorithms are not inherently designed to handle multiple observations simultaneously, making it difficult to effectively integrate the information from each measurement.

To address this limitation, we propose an extension of the existing algorithms to allow for the simultaneous processing of multiple EDM observations. By extending the algorithms in this manner, we can leverage the information from each observation to produce a more robust and accurate estimate of the underlying true EDM. This approach enables us to exploit the redundancy present in multiple measurements, leading to improved performance and reliability in various applications, such as sensor network localization, molecular conformation analysis, and dimensionality reduction tasks (These tasks aim to transform high-dimensional data into a lower-dimensional representation while minimizing the loss of important information [44]).

Chapter 3

Algorithms for processing multiple EDM observations

In our research, we are focused on addressing the challenge of handling multiple observations of EDMs that may suffer from noise and incompleteness. These observations are derived from the true underlying EDM, with each observation subject to its own unique random noise and symmetrically missing entries. Our primary objective is to accurately recover the true EDM despite the presence of noise and missing data. By jointly processing these observations, we anticipate achieving results that are either equivalent to or better than those obtained from processing each observation individually.

To accomplish this, we have developed extensions to the rank alternation and SDR algorithms. These extensions are tailored for handling multiple EDM observations. In our notation, $\widetilde{\mathbf{D}}_k$ represents the k-th incomplete and noisy observed EDM, while $\mathbf{W}_k$ corresponds to its binary mask matrix, indicating the missing entries. These algorithms are designed to effectively exploit the information contained within multiple observations to improve the accuracy of EDM recovery, despite the challenges posed by noise and missing data.

3.1 Extension of Rank-based Algorithm

To extend the rank-based algorithm, we exploit the relationship between the rank-r approximation of a matrix and the Frobenius norm. For a square matrix $\mathbf{A}$ of size $n \times n$, its singular value decomposition (SVD) is expressed as

$$\mathbf{A} = \sum_{i=1}^{n} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{3.1}$$

where the singular values $\sigma_i \geq 0$ are in decreasing order of magnitude, i.e., $\sigma_1 \geq \ldots \geq \sigma_n$. Here, $\mathbf{u}_i$ and $\mathbf{v}_i$ are the left and right singular vectors associated with the singular value σ_i, respectively. The optimal rank-r approximation $\mathbf{A}_r$ of $\mathbf{A}$, minimizing the Frobenius norm $||\mathbf{A} - \mathbf{A}_r||_F$, is

$$\mathbf{A}_r = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{3.2}$$

for $r < n$ (according to matrix approximation lemma or Eckart–Young–Mirsky theorem [45], [46]) [47–50].

This theorem underpins the extraction of a low-rank approximation of $\mathbf{A}$. By retaining the dominant singular values and their corresponding singular vectors, the rank-r approximation $\mathbf{A}_r$ captures the most significant components of the original matrix $\mathbf{A}$. These approximations are widely used in tasks like dimensionality reduction [51–53], signal processing [54–56], and machine learning [57–60]. Here, balancing computational efficiency with the retention of crucial information is important.

The reduced rank approximation achieved through singular value truncation is effectively a minimization of the Frobenius norm. Inspired by this observation, we devise a strategy for rank reduction of multiple input EDMs (K input EDMs in our case), denoted as $\widetilde{\mathbf{D}}_k$ with $\mathbf{W}_k$ being its corresponding mask matrix. Our aim is to recover the true EDM, $\mathbf{D}$, by formulating the following cost.

$$\underset{\mathbf{D}\in\mathbb{R}^{n\times n}}{\text{minimize}} \sum_{k=1}^{K} \left\| \mathbf{W}_k \circ (\mathbf{D} - \widetilde{\mathbf{D}}_k) \right\|_F . \tag{3.3}$$

This cost function indicates the dissimilarity between the multiple observed EDM inputs and the EDM estimate. So the minimization of this cost function means that we are trying to obtain a closer estimate to that of multiple observed EDM inputs. We implement the minimization process using the CVX toolbox in MATLAB. Once an estimate of the EDM is obtained, we enforce the rank property to ensure that its rank does not exceed $d + 2$, where d is the dimensionality. We call this methodology as the ***Joint Rank*** algorithm. Its MATLAB/CVX implementation can be found in algorithm 3.

Algorithm 3 Joint Rank Algorithm

function `JointRank` $((\widetilde{\mathbf{D}_k})_{k=1}^{K}, (\mathbf{W}_k)_{k=1}^{K}, \lambda, d)$
  $n \leftarrow \text{size}(\widetilde{\mathbf{D}_1}, 2)$
  cvx_begin sdp : Begin of CVX implementation
    variable $\mathbf{D}(n, n)$ symmetric : Defining EDM variable
    minimize $\sum_{k=1}^{K} \text{norm}(\mathbf{W}_k \circ (\mathbf{D} - \widetilde{\mathbf{D}_k}), \text{'fro'})$: Defining Optimization problem
  cvx_end : End of CVX implementation
  $\mathbf{D} \leftarrow \text{RankEnforce}(\mathbf{D}, d + 2)$: Enforce rank property
  return $\mathbf{D}$
end function

function `RankEnforce` $(\mathbf{D}, d)$
  $\mathbf{U}, [\lambda_i]_{i=1}^{n}, \mathbf{R} \leftarrow \text{SVD}(\mathbf{D})$
  $\widetilde{\mathbf{P}} \leftarrow \text{diag}(\lambda_1, \ldots, \lambda_d, \underbrace{0, \ldots, 0}_{n-d \text{ zeroes}})$
  $\hat{\mathbf{D}} \leftarrow \mathbf{U}\widetilde{\mathbf{P}}\mathbf{R}^T$
  return $\hat{\mathbf{D}}$
end function

3.2 Extension of SDR Algorithm

The extension of the Semi-Definite Relaxation (SDR) algorithm involves incorporating a cost function that considers all available input EDMs. In other words, the algorithm seeks to minimize a cost function that takes into account the discrepancies between the observed distances in the input EDMs and the corresponding distances implied by the estimated solution.

This extension addresses the issue arising with the existing SDR algorithm, i.e., its inability to process multiple available input EDMs and to obtain the true underlying EDM, as different input EDMs yield different results. Thus, this extension enhances the algorithm's ability to handle various types of input data and improves the quality of the final solution obtained. The resulting optimization problem for this algorithm is given by,

$$\begin{aligned} \max_{\mathbf{H}} \quad & \sum_{k=1}^{K} \Big(\text{tr}(\mathbf{H}) - \lambda \| \mathbf{W}_k \circ (\widetilde{\mathbf{D}_k} - \mathcal{K}(\mathbf{V}\mathbf{H}\mathbf{V}^T)) \|_F \Big) \\ \text{subject to} \quad & \mathbf{H} \in \mathbb{S}_+^{n-1} \end{aligned} \tag{3.4}$$

where $\text{tr}(\mathbf{H})$ represents the trace of the reduced Gram matrix $\mathbf{H}$. This modification of using reduced Gram matrix in the cost function addresses the issue arising from the null space of the Gram matrix $\mathbf{G}$ [39], enabling the acquisition of strictly positive semi-definite solutions. Once $\mathbf{H}$ is solved for and its solution obtained, the Gram matrix $\mathbf{G}$ is computed using its invertible transformation equation, as specified by (2.19). Subsequently, the corresponding EDM can be derived using the Equation (2.4).

Moreover, the choice of λ in the Equation (3.4) is set to 100. We determined this value after conducting numerous experiments to find the optimal λ, testing values in the range of [0.1, 1000]. This approach, termed the ***Joint SDR*** algorithm, allows for the simultaneous processing of all input noisy and incomplete EDMs to obtain their true underlying EDM. Its MATLAB/CVX implementation can be found in Algorithm 4.

3.3 Simulation Results

In order to thoroughly evaluate and compare the performance of proposed joint algorithms against existing algorithms designed for single EDM inputs, it is imperative to conduct rigorous testing through a series of simulations. These simulations serve as invaluable tools in providing insights into the effectiveness of the proposed joint algorithms across various scenarios. By subjecting the proposed algorithms to various simulated conditions, we can gain a comprehensive understanding of their performance and ascertain their potential advantages over existing approaches.

To facilitate comparisons, the subsequent subsections will delve into the methodologies employed for evaluating the proposed joint algorithms alongside their single-input counterparts. This will involve outlining the specific metrics utilized to evaluate performance, such as relative error and input error.

Algorithm 4 Joint SDR Algorithm

function `JointSDR` $((\widetilde{\mathbf{D}_k})_{k=1}^K, (\mathbf{W}_k)_{k=1}^K, \lambda)$

$n \leftarrow \text{size}(\widetilde{\mathbf{D}_1}, 2)$

$x \leftarrow -1/\left(n + \text{sqrt}(n)\right)$

$y \leftarrow -1/\text{sqrt}(n)$

$\mathbf{V} \leftarrow [y * \text{ones}(1, n-1);\ x * \text{ones}(n-1) + \text{eye}(n-1)]$

$\mathbf{e} \leftarrow \text{ones}(n, 1)$

cvx_begin sdp : Begin of CVX implementation

variable $\mathbf{H}(n-1, n-1)$ symmetric : Defining reduced Gram matrix

$\mathbf{G} \leftarrow \mathbf{VHV}^T$

$\mathbf{C} \leftarrow \text{diag}(\mathbf{G}) * \mathbf{e}' + \mathbf{e} * \text{diag}(\mathbf{G})' - 2 * \mathbf{G}$

maximize $\sum_{k=1}^K \left(\text{trace}(\mathbf{H}) \ - \lambda * \text{norm}\left(\mathbf{W}_k \circ (\mathbf{C} - \widetilde{\mathbf{D}_k}), \text{'fro'}\right)\right)$

subject to : Defining Optimization problem

$\mathbf{H} \geq 0$

cvx_end : End of CVX implementation

$\mathbf{G} \leftarrow \mathbf{VHV}^T$: Constructing Gram matrix

$\mathbf{D} \leftarrow \text{diag}(\mathbf{G}) * \mathbf{e}' + \mathbf{e} * \text{diag}(\mathbf{G})' - 2 * \mathbf{G}$: Constructing EDM

return $\mathbf{D}$

end function

Additionally, the types of noise introduced in the simulations will be discussed. Furthermore, the parameters used for comparing the joint algorithms and existing ones will be detailed.

3.3.1 Algorithms

To compare the proposed joint algorithms, which are adept at processing multiple EDM inputs, with the existing algorithms tailored for single EDM input, we'll thoroughly analyze various algorithmic approaches. Our evaluation encompasses scenarios involving multiple EDM inputs. Specifically, we scrutinize joint algorithms that efficiently handle all inputs simultaneously, contrasting them with two distinct variations observed in existing algorithms. This exploration will be detailed further.

These algorithms are: Joint Rank, Rank (Individual), Rank (Average), Joint SDR, SDR (Individual), and SDR (Average). Joint Rank and Joint SDR are discussed in Section 3.1 and Section 3.2 respectively. In Rank (Individual), the rank alternation algorithm is applied individually to each of the input EDM and the average error is reported. Alternately, in Rank (Average), the input EDM are averaged to obtain a single EDM on which the rank alternation algorithm is applied. The average of all input EDM is computed element-wise. The ij^{th} entry of the resulting average EDM is the mean of all known ij^{th} entries in the input EDM. If there is no known ij^{th} entry in any of the input EDM, the corresponding entry in the average EDM is set to 0. Similarly SDR (Individual) and SDR (Average) algorithms are applied.

3.3.2 Error metric

Once the EDM is estimated using algorithm (either joint proposed algorithms or existing algorithms), we need a metric to compare their performance at recovering true underlying EDM. This metric should accurately measure how closely the algorithms estimate the true underlying EDM. To achieve this, we report the average relative error between the true and estimated EDM. The relative error (ϵ) in percentage for one simulation is computed as follows:

$$\epsilon = \frac{\|\mathbf{D} - \hat{\mathbf{D}}\|_F}{\|\mathbf{D}\|_F} \times 100\,, \tag{3.5}$$

where $\mathbf{D}$ represents the true EDM and $\hat{\mathbf{D}}$ denotes the estimated EDM.

In our study, we conducted $20 \times 20 = 400$ simulations aimed at observing a broad performance trend exhibited by both the proposed joint algorithms and the existing algorithms. These simulations were carried out across 20 distinct randomly generated point sets, each subject to 20 different noise realizations. Moreover, the points within these point sets were assumed to reside in the $x - y$ plane, with their coordinates sampled uniformly random in the interval $[0, 10]$.

Following these simulations, we computed the average relative error over 400 runs. Furthermore, for comparison and reference, we also plotted the average input error (Input) observed across these 400 simulations. The input error (ϵ_{in}) for one simulation is given by,

$$\epsilon_{\text{in}} = \frac{1}{K} \times \sum_{k=1}^{K} \frac{\left\|\mathbf{W}_k \circ (\mathbf{D} - \tilde{\mathbf{D}}_k)\right\|_F}{\|\mathbf{W}_k \circ \mathbf{D}\|_F} \times 100 \tag{3.6}$$

which is the average of the relative errors for all the input EDM. The above error percentages are not restricted to values ≤ 100 but provide a good indicator for performance of algorithms. These metrics allow us to analyze and compare the performance of different algorithms under various scenarios.

3.3.3 Noise model

Consider a set of n points forming a point set, with $\mathbf{D}$ representing the EDM that captures the pairwise distances between these points. Now, suppose there are K noisy and incomplete observations of the true underlying EDM, $\mathbf{D}$. Let the signal-to-noise ratio be s. It quantifies the strength of the signal (the true underlying distance matrix) relative to the level of noise present in the observations. A higher s indicates a stronger signal relative to noise.

Additionally, let's assume that $p\%$ of the total entries in each observation are missing. These missing entries are randomly chosen, meaning there's no specific pattern to where the missing values occur within the observations. For the noise model, we consider multiplicative noise, as done in the literature [37], [61–65].

Let $\mathbf{E}$ be the matrix containing distances between n points (EDM entries are squared distances). The noise corrupted matrix is given by,

$$\widetilde{\mathbf{E}} = \mathbf{E} + \mathbf{E} \circ \widetilde{\mathbf{N}} \tag{3.7}$$

where $\widetilde{\mathbf{N}}$ is the $n \times n$ symmetric noise matrix constructed from the noise matrix $\mathbf{N}$ as follows.

$$\widetilde{\mathbf{N}} = \frac{1}{\sqrt{2}} \left(\mathbf{N} + \mathbf{N}^\top \right) \tag{3.8}$$

where the noise matrix $\mathbf{N}$ has entries along its diagonal as zeroes, indicating that the distances between each point and itself are zeroes. The non-diagonal entries of $\mathbf{N}$ follow an independent and identically distributed (i.i.d.) uniform distribution between a range, $[-\frac{\sigma}{2}, \frac{\sigma}{2}]$. This means that the distance measurements between distinct points can vary within the range of $[-\frac{\sigma}{2}, \frac{\sigma}{2}]$.

We have primarily considered multiplicative uniform noise in our simulations because, with multiplicative noise, the noise is highly variable across the elements in the input EDM (as it depends on the elements of the distance matrix).

The noisy (yet complete) EDM is denoted by $\widetilde{\mathbf{D}}$ and is calculated as the element-wise multiplication ($\circ$) of the matrix $\widetilde{\mathbf{E}}$ with itself, i.e., $\widetilde{\mathbf{D}} = \widetilde{\mathbf{E}} \circ \widetilde{\mathbf{E}}$. $p\%$ of the entries in $\widetilde{\mathbf{D}}$ are randomly and symmetrically chosen from both the upper and lower parts of the EDM and made missing, effectively introducing incompleteness into the matrix.

The SNR corresponding to our case (multiplicative noise) is as:

$$s = 10 \log_{10} \left(\frac{1}{\sigma^2} \right) \tag{3.9}$$

We also consider two additional types of random noise: zero-mean Gaussian random noise which is a commonly used noise model, and zero-mean Laplacian random noise due to its heavier tail characteristic. Estimators that use the Laplacian noise model are less likely to be influenced by extreme values, leading to more reliable estimates in the presence of outliers.

For the Gaussian noise, it has a variance of σ^2 , meaning the spread or dispersion of the noise follows a Gaussian distribution with a mean of zero. Similarly, for the Laplacian noise, it also has a variance of σ^2 , indicating that the spread or dispersion of the noise follows a Laplacian distribution with a mean of zero. However, the noise variance for the uniform distribution is $\sigma^2/12$.

We analyzed different noise models under various SNRs, but the emphasis was not on comparing the different noise models. The actual focus was on testing the proposed methods with respect to different parameters and comparing their performance with existing methods. In this process, we conducted the analysis for different noise models to observe the trends in those cases. These noises contribute to the overall variability or uncertainty in the observations, affecting the accuracy and reliability of the data.

3.3.4 Performance analysis of algorithms

In this section, we analyze the performance of the EDM recovery algorithms we discussed. We evaluate their effectiveness across different parameters to gain insights into their capabilities and limitations. Table 3.1 provides a summary of the parameter settings used in different simulations.

We specifically examine algorithm performance concerning several factors: the number of points (n) in the point set (where the EDM size is $n \times n$), the number of input EDMs (K), the SNR (s), and the percentage of missing entries (p). Additionally, we analyze how these algorithms perform across different noise realizations in this section.

Analysis	n	K	SNR	p
ϵ vs. n	$4-20$	4	5	30
ϵ vs. K	20	$2-10$	-5	20
ϵ vs. s	20	4	$-5-35$	30
ϵ vs. p	20	4	-5	$0-90$

Table 3.1 Values of parameters used for different simulations.

3.3.4.1 Number of points (n)

In this subsection, we investigate the impact of varying the number of points (n) in a point set on the accuracy of EDM denoising and completion. To facilitate this analysis, we adhere to the experimental setup detailed in Table 3.1, maintaining the number of EDM inputs at a constant value of $K = 4$, the SNR of the model at $s = 5$ dB, and the percentage of missing entries at $p = 30\%$. The noise model employed in this context is uniform multiplicative noise. By systematically altering the number of points while keeping other parameters constant, we aim to gain insights into how the density of points influences the performance of denoising and completion algorithms.

We plotted the average error in Figures 3.1 and 3.2 as a function of the number of points (n) in the point set. For this purpose, we considered the number of points ranging from 4 to 20. We conducted 400 simulations. These plots correspond to the Rank and SDR algorithms, respectively.

At the SNR value we considered, especially with 30% of the information missing in the input EDMs, it is challenging for the Rank Alternation algorithm to enhance the EDM inputs. From Figure 3.1, it is evident that for most of the choices of n, the Rank (individual) fails to enhance the EDM inputs. However, in the case of Rank (Average), the performance is notably superior as it is improving the EDM inputs to a significant extent. This improvement stems from its ability to estimate the true underlying EDM with greater accuracy by averaging out the EDM inputs based on the positions of its known entries, thereby reducing the percentage of missing entries. Now, the joint rank algorithm we proposed performs similarly or even better than the Rank (Average) approach. Notably, at higher numbers of points, the Joint Rank approach outperforms the Rank (Average) approach.

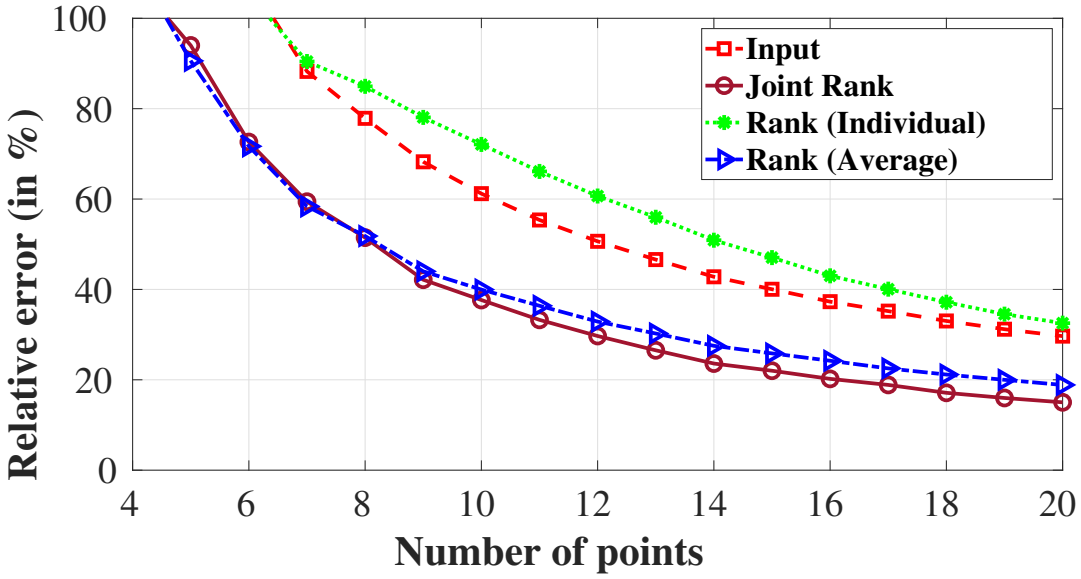

Figure 3.1 Rank: Relative error as function of number of points n.

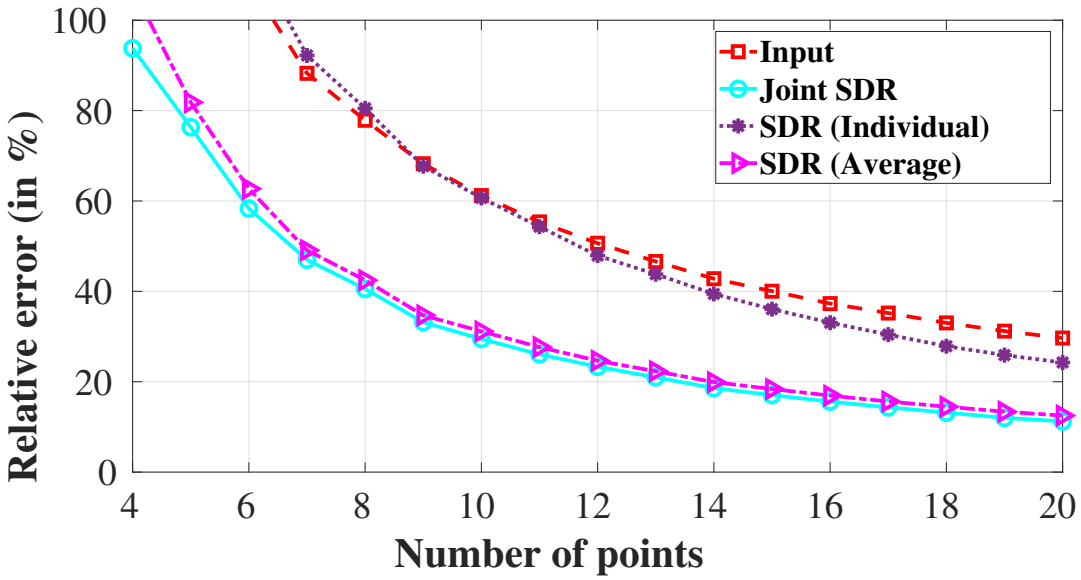

Figure 3.2 SDR: Relative error as function of number of points n.

In a similar setting, the ability of the SDR (Individual) approach to enhance the EDM inputs improves as we increase the number of points in the point set. However, the SDR (Average) approach and the Joint SDR approach perform much better than the SDR (Individual) approach. Both can enhance the EDM inputs effectively. Their performances are similar to each other, but the Joint SDR performs slightly

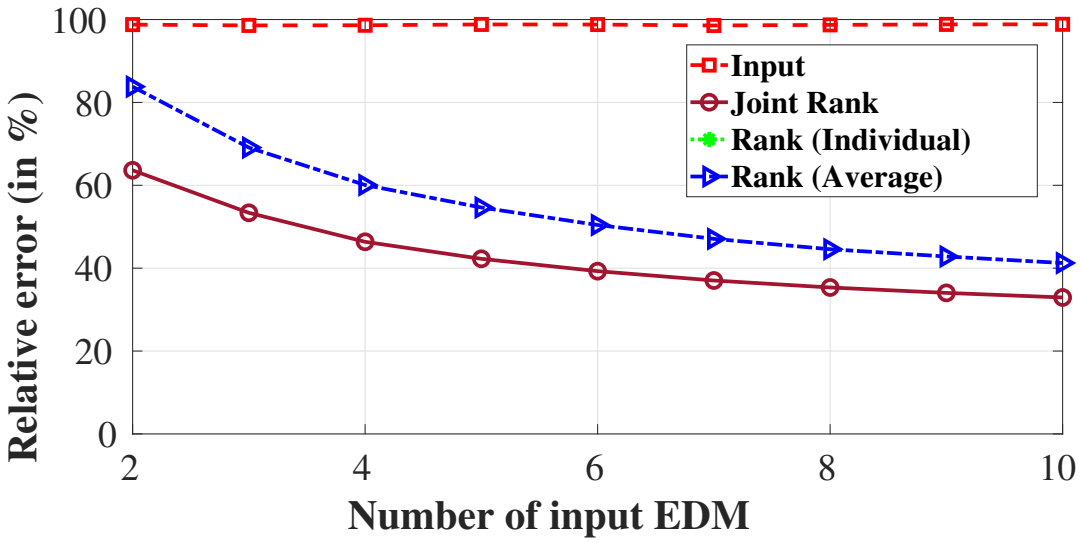

Figure 3.3 Rank: Relative error as function of number of input EDM K.

better than the SDR (Average) for the case of a lower number of points. This analysis indicates that the Joint algorithms perform better than the (Average) and (Individual) methods across various numbers of points considered in the point set.

3.3.4.2 Number of inputs (K)

In this analysis, we vary the number of EDM inputs (K) to the algorithm while fixing the number of points at 20, the signal-to-noise ratio at -5dB, and the percentage of missing entries at 20%, as mentioned in Table 3.1. By comparing plots 3.3 and 3.4, we can observe that SDR methods generally work better compared to Rank methods.

The Rank (Individual) method failed to enhance the EDM inputs. Its error exceeds 100% and hence is not visible in the plot 3.3. The Rank (Average) method is performing much better than Rank (Individual) method, and the Joint Rank method works significantly better compared to Rank Alternation method (both Individual and Average). This is highlighting the benefits of developing algorithms which can jointly process multiple EDMs.

From Figure 3.4, we can observe that both the Joint SDR and SDR (Average) methods have similar performance, with the Joint SDR method being slightly better. For these algorithms, the error reduces as the number of EDMs increases. Additionally, the error of the Individual method does not change with K, as it does not combine the multiple EDMs in any way.

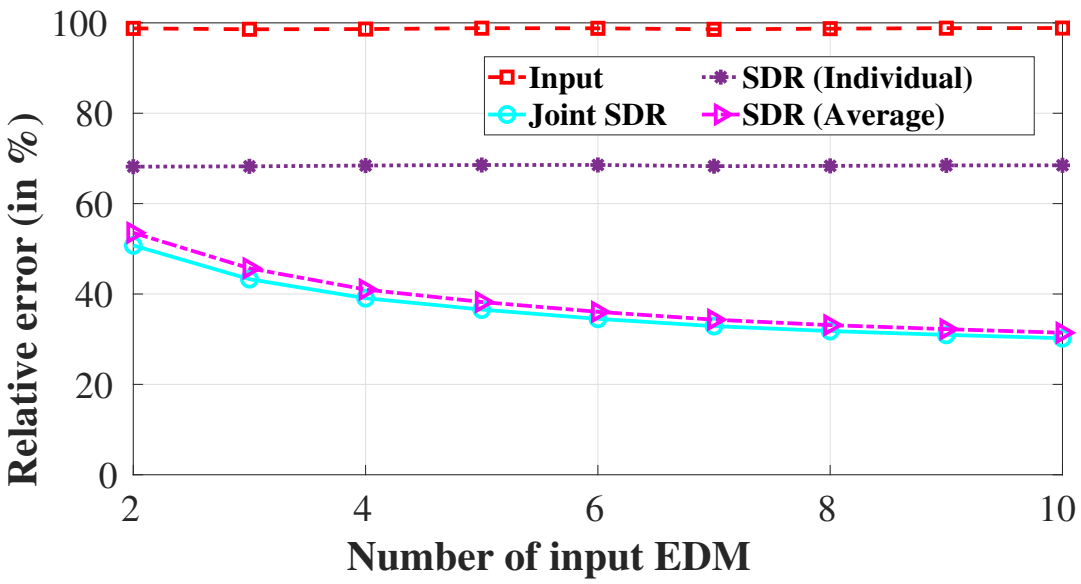

Figure 3.4 SDR: Relative error as function of number of input EDM K.

3.3.4.3 Signal-to-Noise Ratio

The effect of changing the SNR is shown in the figures 3.5 and 3.6, for the setting described in table 3.1 (number of points fixed at 20, number of EDM inputs fixed at 4, and the percentage of missing entries in EDM fixed at 30%). These plots corresponds to the Rank and SDR algorithms respectively.

At low SNR values (< 5 dB), we can observe from Figure 3.5 that the Joint Rank algorithm performs better than other rank-based methods. Whereas at high SNR values, the performance of the Joint Rank algorithm saturates. Again, we can observe that Rank (Individual) is not enhancing the EDM inputs. Also, the same trend is followed here, with Joint SDR and SDR (Average) performing similarly to each other, with Joint SDR slightly outperforming SDR (Average).

One can observe from Figures 3.5 and 3.6 that both the Joint SDR and SDR (Average) outperform the Joint Rank algorithm. Except for Joint Rank, the error of all the algorithms approaches zero as the SNR increases. Since SDR is based on a more detailed modeling of the EDM compared to the simple low-rank characterization used by Rank-based methods, SDR methods outperform Rank methods. But for the same reason, Rank methods run faster compared to SDR methods. This trade-off highlights the balance between computational efficiency and modeling depth in selecting suitable methods for different applications.

3.3.4.4 Missing Entries

We investigate the recovery performance of various algorithms at a relatively low SNR of -5 dB by setting the number of points at 20 and the number of EDM inputs to 4. As expected, for all algorithms, errors increase as the missing percentage p increases, as seen from Figures 3.7 and 3.8. This is because

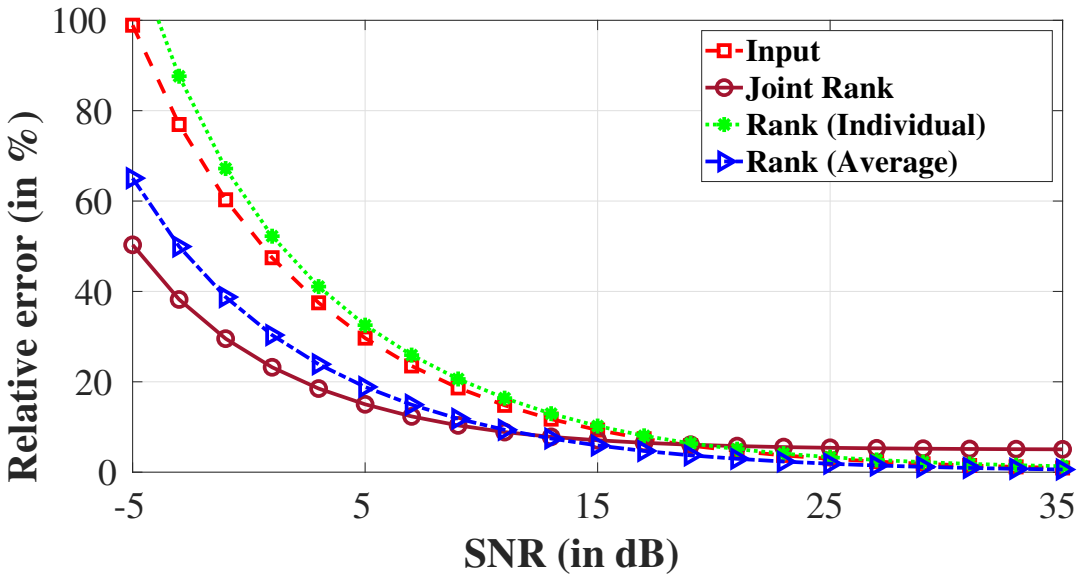

Figure 3.5 Rank: Relative error as function of SNR.

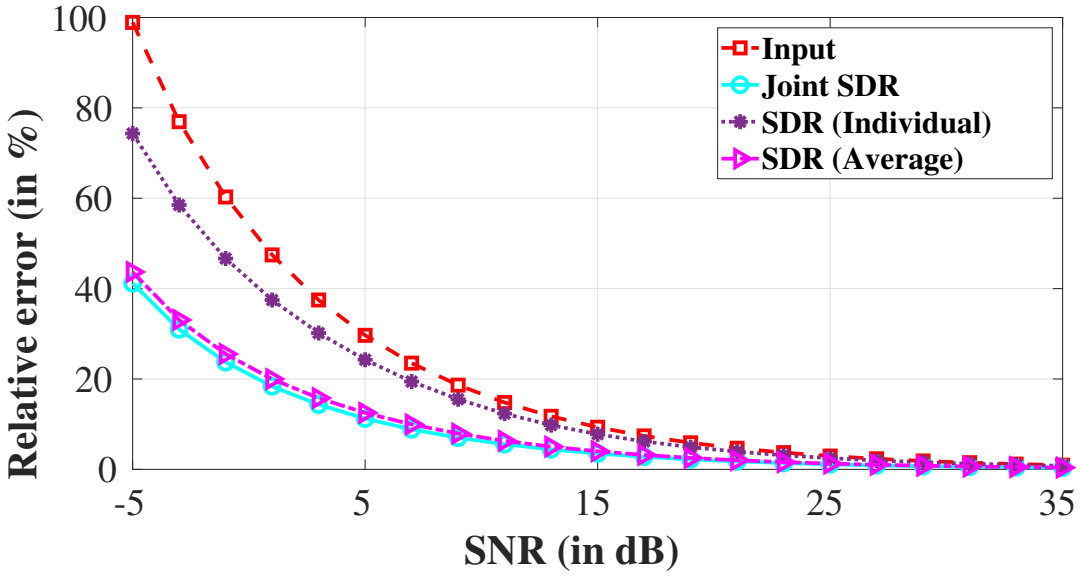

Figure 3.6 SDR: Relative error as function of SNR.

the recovery of the true underlying EDM becomes difficult as the amount of missing data in the EDM inputs increases.

Similar to previous observations, we can observe from Figure 3.7 that the Rank (Individual) approach fails to enhance the EDM inputs, whereas the Rank (Average) approach tries to improve EDM inputs until 70% of the data in EDM inputs is missing. The Joint Rank shows significant improvement over

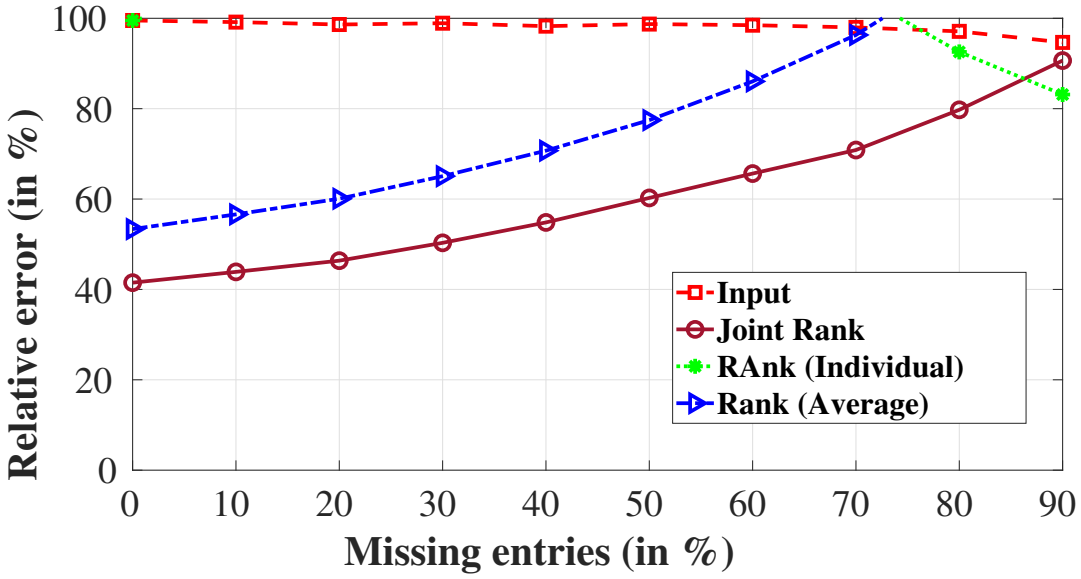

Figure 3.7 Rank: Relative error as function of missing entries (in %) p.

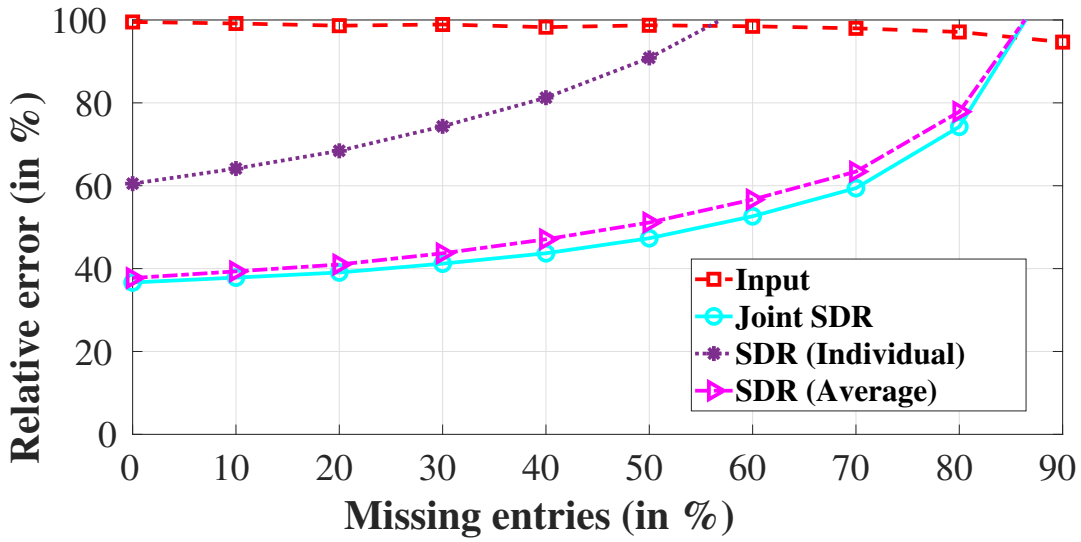

Figure 3.8 SDR: Relative error as function of missing entries (in %) p.

Rank alternation methods (Individual and Average). It can be seen that it can enhance the EDM inputs in all scenarios of missing data, performing better than the Rank (Average) approach.

Also, from Figure 3.8, we can observe that the SDR (Individual) is able to improve upon the input EDM for up to 50% missing entries, beyond which it fails. Whereas the SDR (Average) and Joint SDR are able to improve the input EDM for up to 80% missing entries. The Joint SDR shows a slight improvement (approximately 4% at 50% missing entries) over SDR (Average).

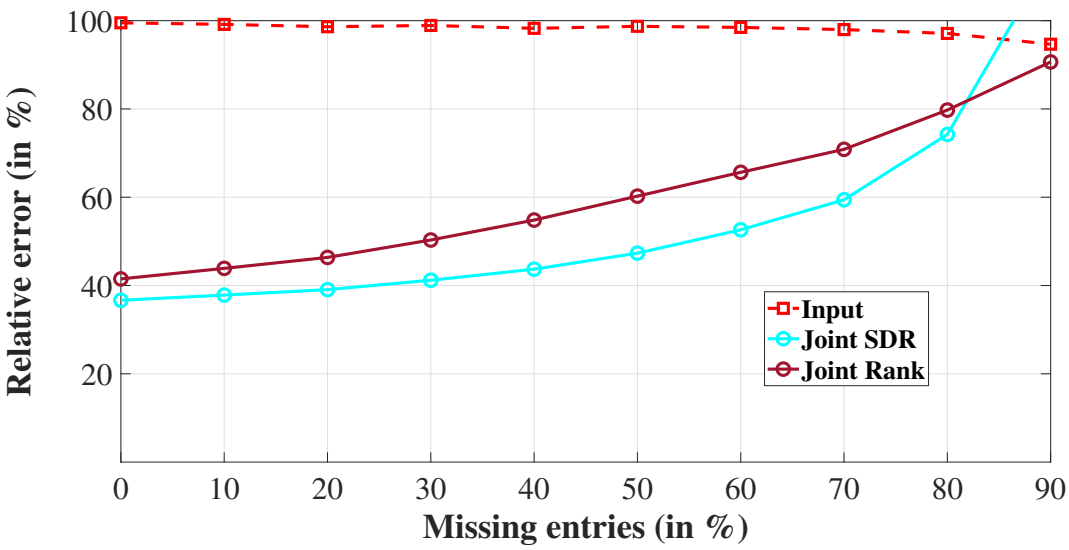

Figure 3.9 Performance comparison of Joint methods.

Comparing both joint algorithms, we can observe from Figure 3.9 that Joint SDR outperforms Joint Rank. The possible reasoning might be that Joint SDR is formed on the basis of more detailed modeling. Additionally, speaking about the runtime of both algorithms, the Joint Rank algorithm takes 18.75 seconds per simulation, whereas the Joint SDR takes 26.25 seconds per simulation. This discrepancy in runtime suggests that the more detailed modeling in Joint SDR, which likely contributes to its superior performance, also incurs a higher computational cost.

3.3.4.5 Complexity comparison

The runtimes of both algorithms are computed for the simulation described in Section 3.3.4.4, following the settings outlined in Table 3.1. The average runtime per simulation is calculated by averaging the runtimes of the 400 simulations considered. The Joint Rank algorithm takes 18.75 seconds per simulation, while the Joint SDR takes 26.25 seconds per simulation. This difference in runtime suggests that the more detailed modeling in Joint SDR, which likely contributes to its superior performance, also results in higher computational costs.

The following analysis is based on our experience with simulations and our understanding of how each of these algorithms works. For both Rank-based and SDR-based algorithms, the Average method has lower time complexity compared to the Joint and Individual methods because it operates on a single input EDM (the matrix obtained after averaging the input EDM).

In the case of SDR, the Individual method has higher time complexity compared to the Joint method. This is because the Individual method uses the CVX solver K times (based on the input size), whereas the Joint method uses it only once. However, the Joint method takes more time than the Average method due to the larger problem size.

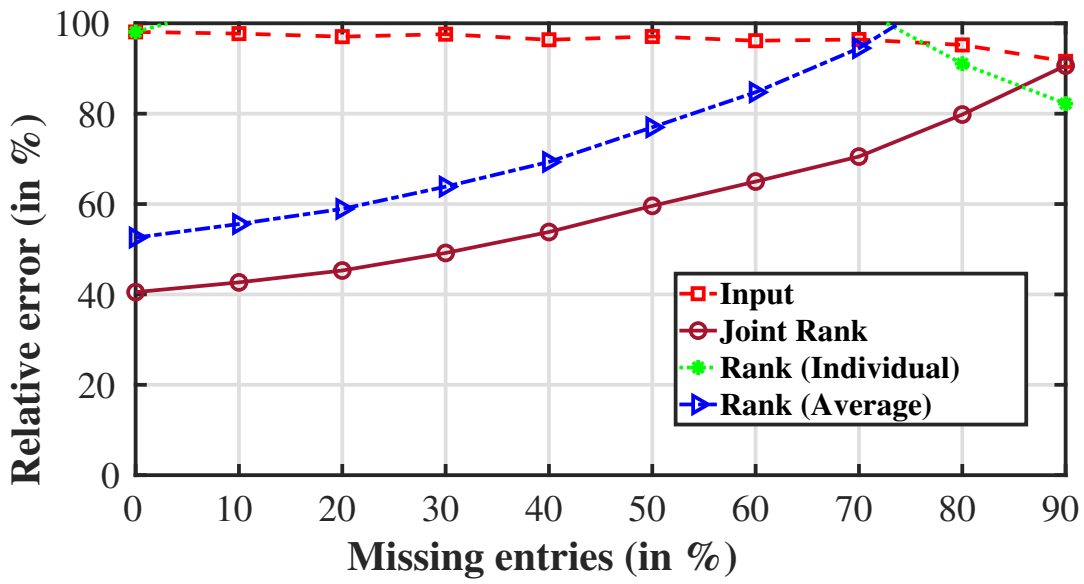

Figure 3.10 Rank (Gaussian multiplicative noise): Relative error as function of missing entries (in %) p.

For Rank-based methods, the Individual methods perform better for smaller input sizes compared to the Joint method. Conversely, for larger input sizes, the Joint method outperforms the Individual method because the iterative approach takes more time compared to the optimization approach.

3.3.4.6 Effect of noise type

We study the effect of noise type on rank and SDR based algorithms. Specifically, we consider zero-mean Gaussian noise and zero-mean Laplacian noise with SNR 6 dB, $n = 20$ points and $K = 4$ input EDM. The missing entries are increased from 0 to 90%.

Results corresponding to rank-based algorithms for Gaussian multiplicative noise are shown in Figures 3.9 and 3.10, and those corresponding to SDR-based algorithms are shown in Figures 3.11 and 3.12 for the same noises, respectively. On comparison with Figures 3.7 and 3.8, we can observe similar trends for zero-mean Gaussian noise and zero-mean Laplacian noise to the ones observed in the presence of zero-mean Uniform noise, for both rank and SDR-based algorithms.

We also note that similar trends are observed for Gaussian and Laplacian noises when other parameters such as the number of points, number of EDM inputs, and SNR value are varied. This consistency in performance across different noise types and parameters underscores the reliability and effectiveness of the proposed algorithms.

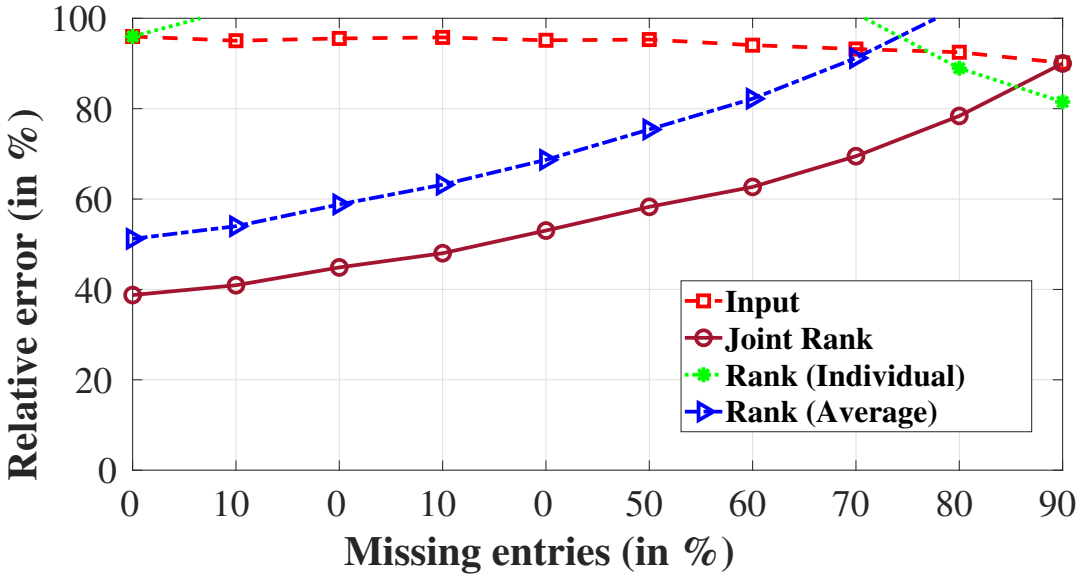

Figure 3.11 Rank (Laplacian multiplicative noise): Relative error as function of missing entries (in %) p.

3.3.4.7 Analysis of Additive Noise Effects

Though our focus is on multiplicative noise, we have also conducted a few experiments on additive noise, where random noise is added to the signal [66]. Just as we discussed the noise model for multiplicative noise in section 3.3.3, we will now discuss the noise model for the additive noise case. For the matrix containing the distances between n points ($\mathbf{E}$) and the $n \times n$ symmetric noise matrix ($\widetilde{\mathbf{N}}$) constructed from the noise matrix $\mathbf{N}$ using Equation (3.8), the noise-corrupted distance matrix is modeled as follows.

$$\widetilde{\mathbf{E}} = \mathbf{E} + \widetilde{\mathbf{N}} \tag{3.10}$$

The noise matrix $\mathbf{N}$ follows the same characteristics as mentioned in section 3.3.3. Additionally, we construct the noisy and incomplete EDM in the same way, as described in section 3.3.3. The type of noise used here is zero-mean uniform random noise with variance of σ^2, and the SNR corresponding to the additive noise case is as follows.

$$s = 10 \log_{10} \left(\frac{\|\mathbf{E}\|_F^2}{n\,(n-1)\,\sigma^2} \right) \tag{3.11}$$

We conducted experiments consisting of 400 simulations, varying the number of points (n). We considered settings with 4 EDM inputs, an SNR value of -10 dB, and 30% of entries missing in the EDM.

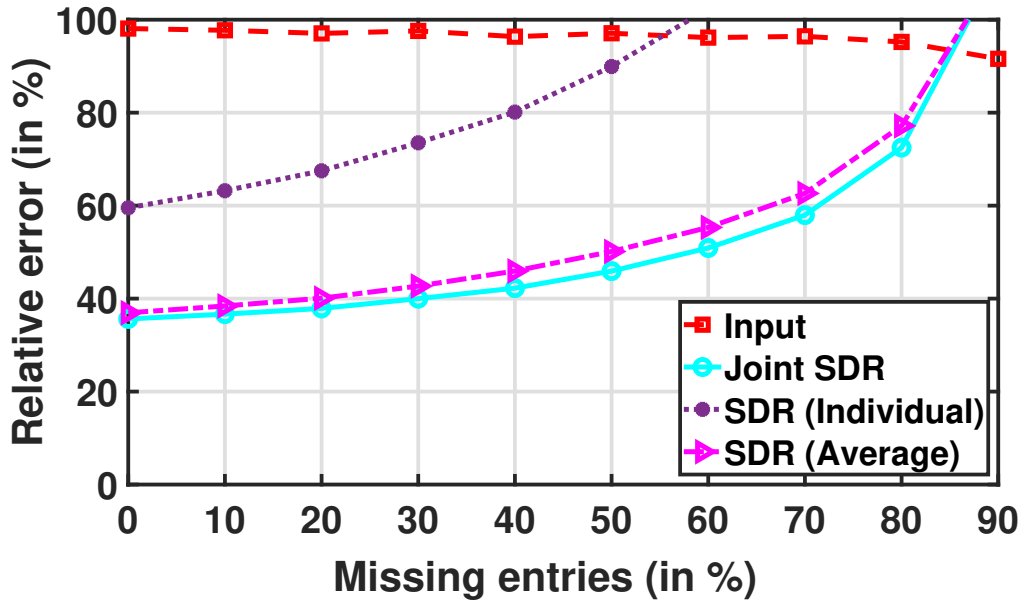

Figure 3.12 SDR (Gaussian multiplicative noise): Relative error as function of missing entries (in %) p.

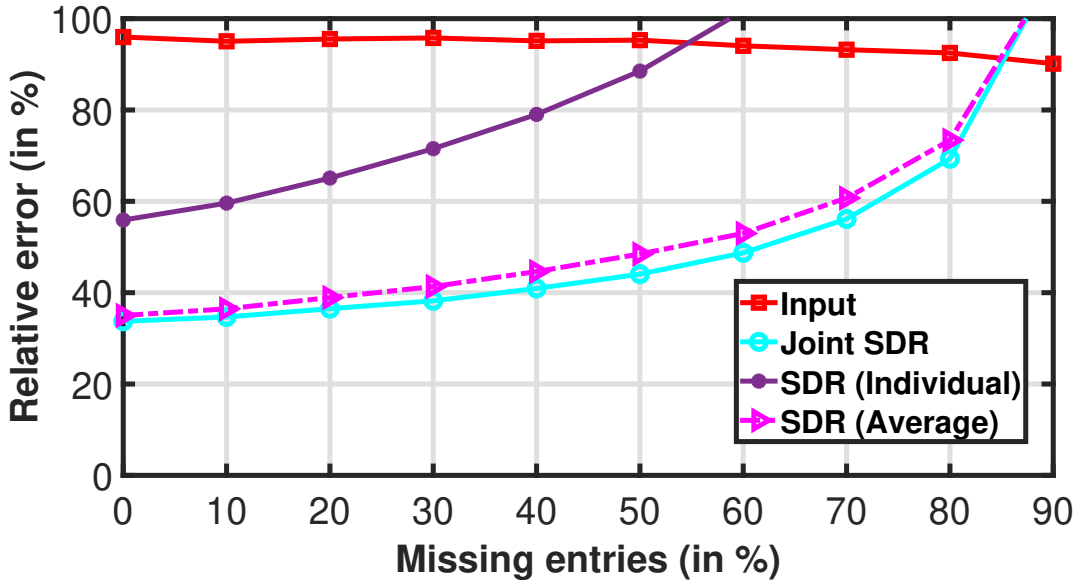

Figure 3.13 SDR (Laplacian multiplicative noise): Relative error as function of missing entries (in %) p.

From Figure 3.13, we can observe that the Rank (Individual) approach fails to obtain a better estimate of the true EDM from the input EDMs. The Rank (Average) approach performs better in estimating the true EDM, but the Joint Rank approach outperforms it significantly as the number of points increases.

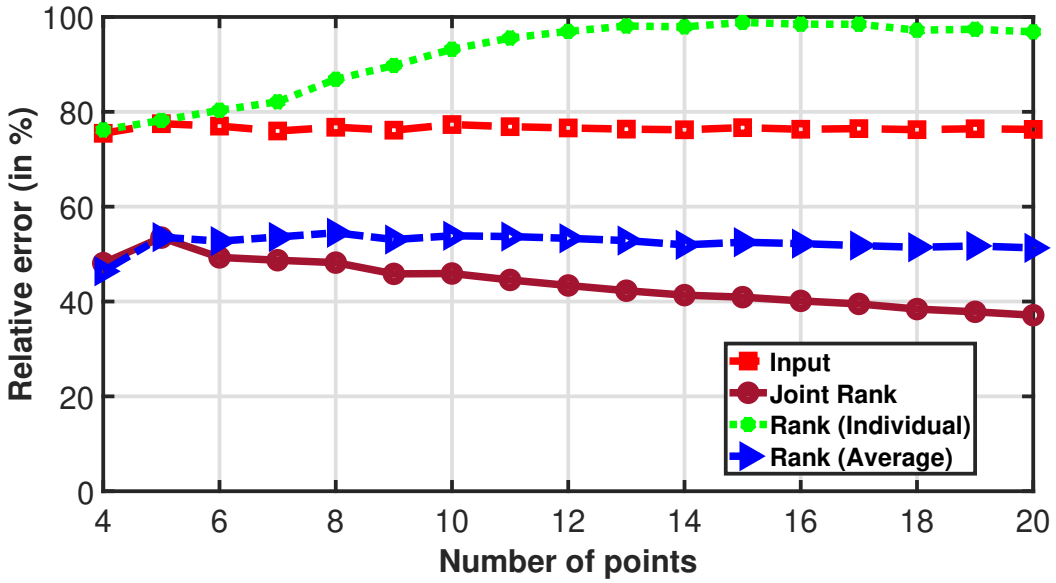

Figure 3.14 Rank: Relative error as function of number of points n, for Additive noise case.

This shows that, in the case of additive noise, the Joint Rank approach is worth considering for estimating the true EDM.

Whereas for the SDR algorithm, more experiments and analysis are needed to study why SDR (Average) still has the upper hand over Joint SDR, while both are trying to improve the input EDMs. It can be noted from Figure 3.14 that the SDR (Individual) approach is failing to improve the input EDM. This summarizes the concise study conducted on the additive noise case.

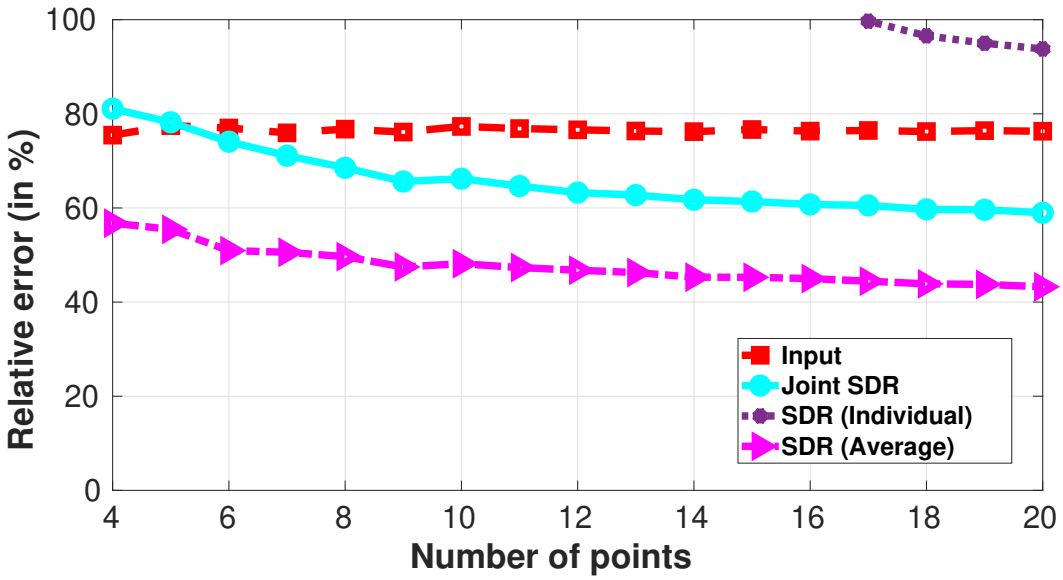

Figure 3.15 SDR: Relative error as function of number of points n, for Additive noise case.

Chapter 4

Conclusion

In this thesis, we introduce novel EDM estimation algorithms designed to handle scenarios where multiple noisy and incomplete EDM observations are provided. Our methods extend the existing SDR algorithm, which is based on mapping between EDMs and Gram matrices, and the Rank alternation algorithm, which is based on the rank priority of an EDM. These algorithms traditionally work with a single input EDM.

The proposed Joint SDR and Joint Rank algorithms demonstrate performance equal to or better than the corresponding baseline algorithms. We verified this through simulations where we varied multiple parameters such as the number of points, the number of EDM inputs, SNR value, amount of missing entries, and noise types such as zero-mean Uniform noise, zero-mean Gaussian noise, and zero-mean Laplacian noise.

Specifically, the Joint Rank algorithm exhibited significant improvement over other rank-based algorithms, while the Joint SDR showed slight improvement over other SDR algorithms. This indicates the effectiveness and robustness of our proposed algorithms in handling noisy and incomplete EDM observations.

A more detailed analysis of the additive noise case for both discussed algorithms could be a potential direction for future work. This idea could also be extended to other EDM denoising and completion algorithms, such as the Alternating Descent algorithm.

Bibliography

[1] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal processing magazine*, vol. 22, no. 4, pp. 54–69, 2005.

[2] K. Menger, "Untersuchungen über allgemeine Metrik," *Mathematische Annalen*, vol. 100, no. 1, pp. 75–163, 1928.

[3] I. J. Schoenberg, "Remarks to maurice frechet's article"sur la definition axiomatique d'une classe d'espace distances vectoriellement applicable sur l'espace de hilbert," *Annals of Mathematics*, pp. 724–732, 1935.

[4] J. Haantjes, "Review: L. M. Blumenthal, Theory and applications of distance geometry," *Bulletin of the American Mathematical Society*, vol. 60, no. 3, pp. 272 – 274, 1954.

[5] G. Young and A. S. Householder, "Discussion of a set of points in terms of their mutual distances," *Psychometrika*, vol. 3, no. 1, pp. 19–22, 1938.

[6] P. Drineas, A. Javed, M. Magdon-Ismail, G. Pandurangan, R. Virrankoski, and A. Savvides, "Distance matrix reconstruction from incomplete distance information for sensor network localization," in *2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, vol. 2. IEEE, 2006, pp. 536–544.

[7] T. F. Havel and K. Wüthrich, "An evaluation of the combined use of nuclear magnetic resonance and distance geometry for the determination of protein conformations in solution," *Journal of molecular biology*, vol. 182, no. 2, pp. 281–294, 1985.

[8] C. Banwell and H. Primas, "On the analysis of high-resolution nuclear magnetic resonance spectra: I. methods of calculating nmr spectra," *Molecular Physics*, vol. 6, no. 3, pp. 225–256, 1963.

[9] V. Bystrov, A. Arseniev, and Y. D. Gavrilov, "Nmr spectroscopy of large peptides and small proteins," *Journal of Magnetic Resonance (1969)*, vol. 30, no. 2, pp. 151–184, 1978.

[10] W. S. Torgerson, "Multidimensional scaling: I. Theory and method," *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952.

[11] K. Weinberger and L. Saul, "Unsupervised learning of image manifolds by semidefinite programming," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, 2004, pp. II–II.

[12] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[13] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean Distance Matrices: Essential theory, algorithms, and applications," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 12–30, 2015.

[14] I. Dokmanić, R. Parhizkar, A. Walther, Y. M. Lu, and M. Vetterli, "Acoustic echoes reveal room shape," *Proceedings of the National Academy of Sciences*, vol. 110, no. 30, pp. 12 186–12 191, 2013.

[15] L. T. Nguyen, J. Kim, S. Kim, and B. Shim, "Localization of IoT networks via low-rank matrix completion," *IEEE Transactions on Communications*, vol. 67, no. 8, pp. 5833–5847, 2019.

[16] R. Parhizkar, "Euclidean distance matrices," Ph.D. dissertation, 2013.

[17] P. Tabaghi, I. Dokmanić, and M. Vetterli, "On the move: Localization with kinetic euclidean distance matrices," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 4893–4897.

[18] R. Kumar, T. Chen, M. Hardt, D. Beymer, K. Brannon, and T. Syeda-Mahmood, "Multiple kernel completion and its application to cardiac disease discrimination," in *2013 IEEE 10th International Symposium on Biomedical Imaging*, 2013, pp. 764–767.

[19] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, 2004.

[20] I. Borg and P. J. Groenen, *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[21] P. H. Schoenemann, *A solution of the orthogonal Procrustes problem with applications to orthogonal and oblique rotation*. University of Illinois at Urbana-Champaign, 1964.

[22] ——, *A solution of the orthogonal Procrustes problem with applications to orthogonal and oblique rotation*. University of Illinois at Urbana-Champaign, 1964.

[23] M. A. Jassim, "The error in calculated distance due to the propagation of positions error," *IOP Conference Series: Materials Science and Engineering*, vol. 737, no. 1, p. 012227, feb 2020. [Online]. Available: https://dx.doi.org/10.1088/1757-899X/737/1/012227

[24] X. Hu, B. Xu, and Y. H. Hu, "Target tracking with distance-dependent measurement noise in wireless sensor networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 5200–5203.

[25] F. A. Monteiro and I. J. Wassell, "Efficient scalar quantization for mimo spatial multiplexing receivers," in *International Symp. on Communication Theory and Applications - ISCTA*, vol. 1, July 2007, pp. —.

[26] E. Eirola, G. Doquire, M. Verleysen, and A. Lendasse, "Distance estimation in numerical data sets with missing values," *Information Sciences*, vol. 240, pp. 115–128, 2013.

[27] W. Glunt, T. L. Hayden, and M. Raydan, "Molecular conformations from distance matrices," *Journal of Computational Chemistry*, vol. 14, no. 1, pp. 114–120, 1993.

[28] J. Matthaei, T. Krüger, S. Nowak, and U. Bestmann, "Swarm exploration of unknown areas on mars using slam," 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:52228274

[29] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, E. Kohler, G. Pottie, M. Hansen, and M. Srivastava, "Sensor network data fault types," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, no. 3, pp. 1–29, 2009.

[30] K. Xie, X. Ning, X. Wang, D. Xie, J. Cao, G. Xie, and J. Wen, "Recover corrupted data in sensor networks: A matrix completion solution," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1434–1448, 2016.

[31] M. El-Gayar, H. Soliman *et al.*, "A comparative study of image low level feature extraction algorithms," *Egyptian Informatics Journal*, vol. 14, no. 2, pp. 175–181, 2013.

[32] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from a few entries," *IEEE Transactions on Information Theory*, vol. 56, no. 6, pp. 2980–2998, 2010.

[33] H. Maehara, "Euclidean embeddings of finite metric spaces," *Discrete Mathematics*, vol. 313, no. 23, pp. 2848–2856, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0012365X13003841

[34] J. C. Gower, "Euclidean distance geometry," *Math. Sci*, vol. 7, no. 1, pp. 1–14, 1982.

[35] ——, "Properties of Euclidean and non-Euclidean distance matrices," *Linear algebra and its applications*, vol. 67, pp. 81–97, 1985.

[36] A. Y. Alfakih, A. Khandani, and H. Wolkowicz, "Solving euclidean distance matrix completion problems via semidefinite programming," *Computational optimization and applications*, vol. 12, pp. 13–30, 1999.

[37] P. Biswas, T.-C. Liang, K.-C. Toh, Y. Ye, and T.-C. Wang, "Semidefinite programming approaches for sensor network localization with noisy distance measurements," *IEEE transactions on automation science and engineering*, vol. 3, no. 4, pp. 360–371, 2006.

[38] V. S. Mai, D. Maity, B. Ramasubramanian, and M. C. Rotkowitz, "Convex methods for rank-constrained optimization problems," in *2015 Proceedings of the Conference on Control and its Applications*. SIAM, 2015, pp. 123–130.

[39] N. Krislock and H. Wolkowicz, *Euclidean distance matrices and applications*. Springer, 2012.

[40] K. Q. Weinberger and L. K. Saul, "Unsupervised learning of image manifolds by semidefinite programming," *International journal of computer vision*, vol. 70, pp. 77–90, 2006.

[41] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," 2014.

[42] M. C. Grant and S. P. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent advances in learning and control*. Springer, 2008, pp. 95–110.

[43] Y. Takane, F. W. Young, and J. De Leeuw, "Nonmetric individual differences multidimensional scaling: An alternating least squares method with optimal scaling features," *Psychometrika*, vol. 42, pp. 7–67, 1977.

[44] A. Ghodsi, "Dimensionality reduction a short tutorial," *Department of Statistics and Actuarial Science, Univ. of Waterloo, Ontario, Canada*, vol. 37, no. 38, p. 2006, 2006.

[45] E. Schmidt, "Zur Theorie der linearen und nichtlinearen Integralgleichungen," *Mathematische Annalen*, vol. 63, no. 4, pp. 433–476, 1907.

[46] C. Eckart and G. M. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, pp. 211–218, 1936. [Online]. Available: https://api.semanticscholar.org/CorpusID:10163399

[47] J. A. Cadzow, "Signal enhancement - a composite property mapping algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 1, pp. 49–62, 1988.

[48] J. Cadzow and D. Wilkes, "Signal enhancement and the SVD," in *Proceedings of the 2nd International Workshop on SVD and Signal Processing, University of Rhode Island, Kingston, RI*, 1990, pp. 144–151.

[49] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.

[50] P. C. Hansen, "The truncated SVD as a method for regularization," *BIT Numerical Mathematics*, vol. 27, pp. 534–553, 1987.

[51] K. Li and G. Wu, "A randomized generalized low rank approximations of matrices algorithm for high dimensionality reduction and image compression," *Numerical Linear Algebra with Applications*, vol. 28, no. 1, p. e2338, 2021.

[52] M. B. Cohen, S. Elder, C. Musco, C. Musco, and M. Persu, "Dimensionality reduction for k-means clustering and low rank approximation," in *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, 2015, pp. 163–172.

[53] Z. Liu, Y. Lu, Z. Lai, W. Ou, and K. Zhang, "Robust sparse low-rank embedding for image dimension reduction," *Applied Soft Computing*, vol. 113, p. 107907, 2021.

[54] P. Parker, P. J. Wolfe, and V. Tarokh, "A signal processing application of randomized low-rank approximations," in *IEEE/SP 13th Workshop on Statistical Signal Processing, 2005*. IEEE, 2005, pp. 345–350.

[55] M. T. Chu, R. E. Funderlic, and R. J. Plemmons, "Structured low rank approximation," *Linear algebra and its applications*, vol. 366, pp. 157–172, 2003.

[56] I. Markovsky, "Structured low-rank approximation and its applications," *Automatica*, vol. 44, no. 4, pp. 891–909, 2008.

[57] F. Wen, L. Chu, P. Liu, and R. C. Qiu, "A survey on nonconvex regularization-based sparse and low-rank recovery in signal processing, statistics, and machine learning," *IEEE Access*, vol. 6, pp. 69 883–69 906, 2018.

[58] P. Indyk, A. Vakilian, and Y. Yuan, "Learning-based low-rank approximations," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[59] M. Udell, C. Horn, R. Zadeh, S. Boyd *et al.*, "Generalized low rank models," *Foundations and Trends® in Machine Learning*, vol. 9, no. 1, pp. 1–118, 2016.

[60] D. Lee, S. J. Kwon, B. Kim, and G.-Y. Wei, "Learning low-rank approximation for cnns," *arXiv preprint arXiv:1905.10145*, 2019.

[61] M. Vural, C. Yuan, N. Kleppmann, P. Jung, and S. Stańczak, "Illinois-Type Methods for Noisy Euclidean Distance Realization," *IEEE Signal Processing Letters*, vol. 29, pp. 2687–2691, 2022.

[62] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, 2004, pp. 46–54.

[63] S. Kim, M. Kojima, and H. Waki, "Exploiting sparsity in sdp relaxation for sensor network localization," *SIAM Journal on Optimization*, vol. 20, no. 1, pp. 192–215, 2009.

[64] T. K. Pong and P. Tseng, "(robust) edge-based semidefinite programming relaxation of sensor network localization," *Mathematical programming*, vol. 130, no. 2, pp. 321–358, 2011.

[65] Z. Wang, S. Zheng, Y. Ye, and S. Boyd, "Further relaxations of the semidefinite programming approach to sensor network localization," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 655–673, 2008.

[66] M. Fazel, H. Hindi, and S. P. Boyd, "Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices," in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 3. IEEE, 2003, pp. 2156–2162.