THE CATHOLIC UNIVERSITY OF AMERICA

Wireless Sensor Localization With Distance Measurement Information

A DISSERTATION

Submitted to the Faculty of the

Department of Electrical Engineering and Computer Science

School of Engineering

Of The Catholic University of America

In Partial Fulfillment of the Requirements

For the Degree

Doctor of Philosophy

By

Jing Wang

Washington,D.C.

2011

Wireless Sensor Localization With Distance Measurement Information

Jing Wang,Ph.D.

Director: Phillip A. Regalia, Ph.D.

The dissertation would focus on two problems: localization in wireless sensor network and distance reconstruction. Sensors are often randomly deployed and in such case, determining each sensor's position is critical, which explains the recent attention given to the sensor localization problem. Previous methods of projection onto convex sets (POCS) overcome the multimodality problem that plagues earlier least-squares formulations.

Previous efforts in this direction require that the sensor be located in certain position in the plane. Here we propose a new algorithm which projects onto the boundary of convex sets, and features a computationally simple update procedure. The new algorithm has a unique solution no matter where the sensor is located at. Simulation results will be shown and conclusions would be given.

Sensor localization typically exploits distance measurements to infer sensor positions with respect to known anchor nodes. Missing or unreliable measurements for specific nodes can impede such procedures, raising the problem of distance measurement reconstruction using distance information from other nodes. The problem has traditionally been approached through multidimensional scaling, and more recently through semidefinite programming and low-rank matrix completion. Here we develop new iterative reconstruction algorithms which instead exploit inertia of key matrices, thereby encompassing stronger constraints than rank alone. This serves to overcome limitations observed in earlier competitive approaches which do not exploit the problem structure as well. Simulation examples illustrate the performance of the new algorithms.

This dissertation by Jing Wang fulfills the dissertation requirement for the doctoral degree in Electrical Engineering approved by Phillip Regalia, Ph.D., as Director, and by Nader Namazi, Ph.D., Ozlem Kilic, Ph.D. as Readers.

____

Phillip Regalia, Ph.D., Director

____

Nader Namazi, Ph.D., Reader

____

Ozlem Kilic, Ph.D., Reader

Table of Contents

List of Figures

List of Abbreviations

MLE	Maximum Likelihood Estimation
SNR	Signal Noise Ratio
CRB	Cramér-Rao Bound
MDS	Multidimensional Scaling
WLS	Weighted Least Squares
ML	Maximum Likelihood
RSS	Received Signal Strength
TOA	Time of Arrival
AOA	Angle of Arrivel
POCS	Projection Onto Convex Set
NLS	Non-linear Least Squares
RSSI	Received Signal Strength Indicator
EBL	Energy Based source Localization
RMSE	Root Mean Square Error
SDP	Semidefinite Programming

Acknowledgments

I owe my gratitude to all the people who have made this dissertation possible and because of whom my graduate experience here in Catholic University of America has been one that I will cherish forever.

First and foremost I'd like to thank my advisor, Dr. Phillip Regalia for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past 3 years. He gave me precious opportunities to present and learn things in conferences. It has been a pleasure to work with and learn from such an extraordinary individual.

I would also like to thank my professor Dr.Nader Namazi. I have learned a lot from his class which is full of extraordinary theoretical ideas and computational expertise which contributed so much for my thesis. Thanks are due to Dr.Phillip Regalia, Dr.Nader Namazi and Dr.Ozlem Kilic, for agreeing to serve on my thesis committee and for sparing their invaluable time reviewing the proposal and this thesis.

I owe my deepest thanks to my parents who have always stood by me and given me all their support. I also want to thank my friend Christopher Stephens and his family, Tianyue Ma, who have always stood by me during difficult times. I would also like to acknowledge help and support from some of my classmates and friends: Ai Liu, Amy Lin, Ali Basiri, Hamid Karimpour, Kennith White, Pedro Gonzales and Roberto Sliva, Robert Schell. Their help during my life and study here is highly appreciated. I would also like to thank Dr. Robert Meister, Dr. Mohammad Arozullah and Dr.

Scott Matthews who are so friendly and kind to me.

I would like to acknowledge financial support from the National Science Foundation(NSF).

Chapter 1

Introduction Of Wireless Sensor Networks

In this chapter, we introduce wireless sensor networks and their applications, which have received considerable attention recently. A general introduction of wireless sensor networks localization algorithms is also given.

Wireless sensor networks are a new class of distributed systems that are an integral part of the physical space they inhabit [2]. Sensors detect the world's physical nature, such as light intensity, sound, temperature or proximity to objects. Sensor networks have captured the attention of many researchers. Despite their variety, all sensor networks have certain fundamental features in common, including limited computational abilities and two-way radio system.

Sensor networks may be viewed as large collections of nodes. Individually, each node is autonomous and has short range; collectively, they are cooperative and effective over a large area. A system composed of many short-range sensors lends itself to a very different set of applications than one that uses a small number of powerful, long-range sensors.

1.1 Wireless Sensor Networks Application

Potential applications for wireless sensor networks are vast; here some example applications are given.

Military Applications: Wireless sensor networks can be an integral part of military command, control, communications, computers, intelligence, surveillance, reconnaissance and tracking systems. The rapid deployment, self organization and fault tolerance characteristics of sensor networks make them a very promising sensing technique for military. Since sensor networks are based on the dense deployment of disposable and low cost sensor nodes, destruction of some nodes by hostile actions dose not affect a military operation as much as the destruction of a traditional sensor. Some of the military applications are monitoring friendly forces, equipment and ammunition; battlefield surveillance; reconnaissance of opposing forces and terrain; targeting; battle damage assessment; and nuclear, biological and chemical attack detection and reconnaissance.

Environmental Applications: Some environmental applications of sensor networks include tracking the movements of species, i.e., habitat monitoring; monitoring environmental conditions that affect crops and livestock; irrigation; macroinstruments for large-scale Earth monitoring and planetary exploration; and chemical/biological detection.

Commercial Applications: Sensor networks are also applied in many commercial applications. Some of them are building virtual keyboards; managing inventory control;

monitoring product quality; constructing smart office spaces; and environmental control in office buildings.

1.2 Wireless Sensor Networks Localization

Wireless sensor network location refers to the method of finding the geographic coordinates of a sensor node in wireless local area network environments [2].

Relative to other types of distributed systems, distributed sensor systems introduce an interesting new twist: they are coupled to the physical world, and their spatial relationship to other objects in the world is typically an important factor in the task they perform. The term localization refers to the collection of techniques and mechanisms that measure these spatial relationships.

When raw sensor data is combined with spatial information, the value of the data and the capability of the system that collects it increases substantially. For example. a collection of temperature readings without location information is at best only useful to compute simple statistics such as the average temperature. At worst. analysis of the data might yield incorrect conclusions if inaccurate assumptions of the data might yield incorrect conclusions if inaccurate assumptions are made about the distribution of physical sampling. By combining the data with location information, the resulting temperature amp can be analyzed much more effectively.

1.2.1 Ranging Technologies

The mechanisms used to measure physical distances are an important factor when designing a localization system. In this section different types of distance measurement are discussed.

Received Signal Strength. Received Signal Strength is roughly a measure of the amplitude of a detected radio signal at a receiver. Often, Received Signal Strength is equivalently reported as measured power, i.e., the squared magnitude of the signal strength [3]. When the model for path loss is assumed to be a function of distance, the received signal strength should be generally decrease as a function of distance. The path loss model is highly dependent on environmental factors: in open space the model is $1/R^2$; near the ground, the model is closer to $1/R^4$.

Time of Arrival. Time of Arrival is the measured time at which a signal first arrives at a receiver. The measured Time of Arrival is the time of transmission plus a propagation-induced time delay [3]. This time delay, $T_{i,j}$, between transmission at sensor i and reception at sensor j, is equal to the transmitter-receiver separation distance, $d_{i,j}$, divided by the propagation velocity, v_p. This speed for RF is approximately 10^6 times as fast as the speed of sound; as a rule of thumb, for acoustic propagation, 1 ms translates to 1 ft(0.3m), while for RF, 1 ns translates to 1 ft.

Angle of Arrival . By providing information about the direction to neighboring sensors rather than the distance to neighboring sensors, Angle of Arrival measurements provide localization information complementary to the Received Signal Strength and

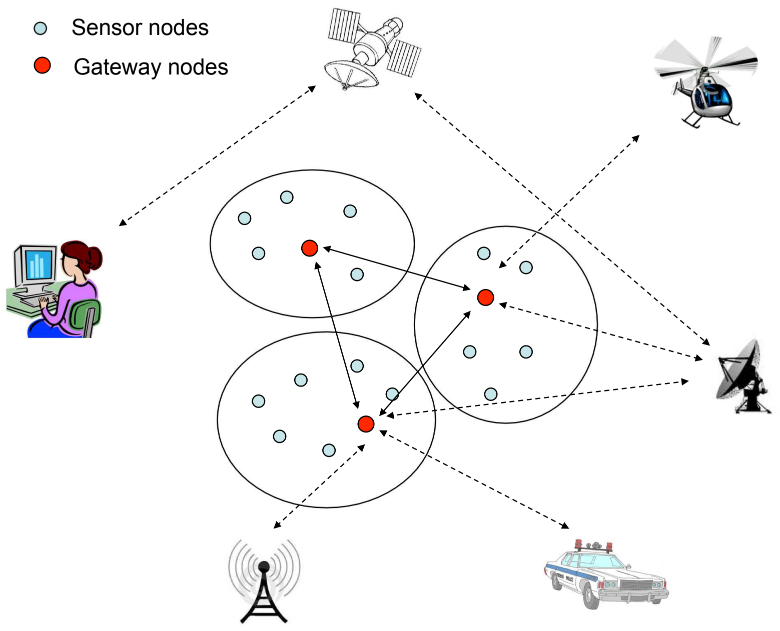

Figure 1.1: Wireless sensor networks

Time of Arrival measurements.

1.2.2 Wireless Sensor Networks Localization Algorithms

Algorithms of wireless sensor networks localization problems can be divided into two categories, each will be followed by some examples:

CENTRALIZED ALGORITHMS

Maximum likelihood estimation method is implemented usually when data is described by a particular statistical model (e.g., Gaussian) [4],[3]. The variance of these estimators asymptotically approaches the lower bound given by the Cramér-Rao Bound , as the Signal Noise Ratio goes high.

The maximum likelihood objective function must be found when maximum likelihood estimation method is used. For sensor localization, however, two difficulties emerge:

- *Local maxima:* The maximum likelihood objective function usually has multiple local maxima. Unless the initial value is close to the global maximum, the search algorithm can be trapped in a suboptimal solution.
- *Model dependency:* When different ranging techniques or model parameters are used in the assumed model, the result can not be guaranteed to be accurate.

One way of preventing local maxima is to formulate the localization as a convex optimization problem [5],[6]. These provide search alternatives of the Projection Onto Convex Set (POCS) approach. However, the Projection onto Convex Set method works well only when the sensor is located in certain area (convex hull) of the network, which will be detailed in chapter 2.

DISTRIBUTED ALGORITHMS

In some applications, no central processor (or none with enough computational power) is available to handle the calculations. Also in a large network, there will be a communication bottleneck and high energy drain near the central processor when all measurement data are forwarded.

Distributed algorithms are developed and employed in many situations. In [7],[8], each sensor estimates its multihop range to the nearest reference nodes. These ranges can be estimated via the shortest path between the sensor and reference nodes, i.e., proportional to the number of hops or the sum of measured ranges along the shortest path. Each node's coordinates are calculated locally via multilateration, when the sensor has range estimates to known positions [9]. Some algorithms use an iterative refinement method. Those algorithms try to find the optimum of a global cost function like WLS or ML. Each node transmite estimated position to its neighbors and the neighbors calculate the position again and transmits to their position again untill convergence is achieved [10].

A new localization method will be discussed in chapter 2.

1.3 Distance Information Reconstruction

Distance is the critical information in wireless sensor network localization. However, specific distance measurements can be obscured due to various factors, including bursty interference or transient jamming during distance measurements, or inadvertent obstacles impeding communication between select sensor pairs. In such cases, the various localization methods may have insufficient input data, necessitating estimation of the missing distance measurements from those available.

In situations when the distance measurements are not precise, one possible method is to find an objective function of a feasible set of consistent distance information set. Given the imprecise distances, the goal is to find the set of distances which minimizes the desired objective functions with respect to the given distances. Using this method [11], [12] determined the location of sensors by lateration and [13] by min-max optimization. In [14], these imprecise distances can be made more accurate and consistent by exploiting fully the geometric and algebraic relations among the distances between nodes.

Also, when the distance information is missing there are many algorithm developed for the reconstruction of the missing or unreliable information. The distance reconstruction problem has considerable geometric structure [15], [16] and we examine further in this work the structural features exploited in [16] based on the low-rank character of a certain matrix constructed from the pairwise distances, which include the Multidimental scaling (MDS) based approaches described in [17][18][19].

Chapter 2

Localization in Wireless Sensor Networks

2.1 Overview

In this chapter we study localization problems in wireless sensor networks. An introduction of traditional method would be given, the shortcoming of which made the development of new algorithms necessary. The next chapter would then mainly focus on our new localization algorithm and its performance.

The emergence of wireless sensor networks has enabled a large number of novel applications which benefit many fields, such as environmental and habitat monitoring and target tracking. In these applications it is necessary to accurately find the position information of the nodes with respect to a global coordinate system so that the reported data is geographically meaningful. The term **localization** refers to the collection of techniques and mechanisms that measure these spatial relationships [2]. In practice, it is infeasible to use expensive GPS to locate each sensor. Instead, each node is low cost with limited power and only local communication with neighbors.

Algorithms discussed here will mainly be based on localization using the received signal strength distance measurement. Received signal strength may be defined as the voltage measured by a receiver's received signal strength indicator circuit [3].

Received signal strength is often equivalently reported as measured power, which is proportional to the squared magnitude of the signal strength. Wireless sensors communicate with neighboring sensors, so the received signal strength can be measured by each receiver during normal data communication without presenting additional bandwidth or energy requirements. Received signal strength measurements are relatively inexpensive and simple to implement in hardware.

2.2 Traditional Algorithm

The problem of locating an acoustic source in a wireless sensor network has been addressed in several articles [20], [21], [1], [22] and references therein. The localization problem in this case has been traditionally solved as a non-linear least squares problem. When the noise is modeled as Gaussian, this is equivalent to maximum likelihood estimation.

Centralized methods, [22] for instance, formulate a maximum likelihood approach. This requires that all the data be transmitted to the fusion center for processing. In relatively large scale networks, the massive amount of energy and bandwidth required make this method impractical. [1],[23] proposed a decentralized implementation of the incremental gradient method, which aims at solving the problem in a distributed manner, thereby alleviating the need to submit massive information to the data center for processing. [24] also proposed an distributed EM algorithm which requires a sufficient number of communication cycles across the network to implement estimator.

However, traditional gradient search methods suffer from searching the global solution in a cost function with local optima and saddle points. [25] proposed a projection onto boundary sets approach that overcome this shortcoming by formulating the problem as a convex feasibility problem instead of traditional nonlinear least squares. Problems may arise when sensors are not within the convex hull of the anchor nodes, which will be discussed later in section 2.3. This method also addresses the localization problem in a distributed manner.

Firstly, however, we review the centralized maximum likelihood approach.

2.2.1 Centralized Maximum Likelihood Approach

Centralized Maximum Likelihood [22] is a source localization approach using acoustic energy measurements from the individual sensors in the sensor field. Maximum Likelihood (ML) estimation is proposed therein to solved this energy based source location problem. In the previous work of [22], the authors showed that in a noiseless situation, each energy ratio dictates that the potential target location must be on a hyper-sphere formed by all energy ratios within the sensor field. With noise taken into account, the target location is solved as the position that is closest to all the hyper-spheres formed by all energy ratios in the least square sense. Using this energy ratio function, the source energy is eliminated, the task of source location estimation can be simplified by solving a nonlinear least square problem.

2.2.1.1 Acoustic Energy Decay Model of Localization Problem

Efficient collaborative signal processing algorithms that consume less energy for computation and communication are important in wireless distributed sensor network communication system [21], which is a significant issue in centralized algorithms. [22] presented a method to estimate the source location based on acoustic energy measured at individual sensors in the sensor network. An acoustic energy decay model is set up in the free and homogenous space under the conditions that the acoustic sources are not far away from the sensors.

Energy-based source localization method is based on the observation that the received sound level decreases when the distance between the source and the listener becomes larger. Thus by estimating the received energy level one can estimate the distance and further estimate the position of each source. There are many factors that may affect sound propagation, such as wind direction, strength of wind, forest or other obstructions. To simplify the problem, there are some assumptions when developing the energy decay model: sound propagates in free air, which is roughly homogenous, and the target is pre-detected to be in a particular region of a sensor field, and the source could be treated as a point. Based on such assumption, the acoustic intensity attenuated at a rate that is inversely proportional to the distance between source and the receiver [26]. As sound waveforms are additive, the acoustic wave intensity signature received by each sensor is [22]:

$$a_i(n) = s_i(n) + \delta_i(n) \tag{2.1}$$

Here consider a sensor network with N sensor nodes, whose positions are denoted as $\mathbf{c}_i \in \mathbb{R}^2$, $i = 1, \ldots, N$. Where $s_i(n) = \gamma_i \Sigma_{k=1}^{K} \frac{a_k(n-t_{ki})}{\|\rho_k(n-t_{ki})-r_i\|}$, $a_i(n)$ is the n-th acoustic signature sampled on the i-th sensor over a time interval $[1/f_s]$ by a matched filter, f_s is the sampling frequency; $\delta_i(n)$ is the zero-mean additive white Gaussian noise (AWGN) on the n-th time interval, K is the number of targets;

Assuming $s_i(n)$ and $\delta_i(n)$ are uncorrelated, different target signal readings are uncorrelated, and $E[\delta_i(n)] = 0$, we have

$$E[a_i^2(n)] = E[s_i^2(n)] + E[\delta_i^2(n)] \tag{2.2}$$

Denoting $E[a_i^2(n)]$ as $y_i(t)$ and $E[\delta_i^2(n)]$ as $\varepsilon_i(t)$, the energy decay model is:

$$y_i(t) = y_s(t) + \varepsilon_i(t) = g_i \sum_{j=1}^{N} \frac{S_j(t)}{\|\mathbf{x}_j(t) - \mathbf{c}_j\|} + \varepsilon_i(t) \tag{2.3}$$

Where $t = \frac{T}{2}, \frac{3T}{2}, \frac{5T}{2}, \ldots$. The background noise $\varepsilon_i(t)$ is independent zero mean AWGN with variance σ_n^2, g_i is a scaling factor corresponding to sensor gain of the i-th acoustic sensor.

2.2.1.2 Maximum Likelihood Estimation for Localization problem

Based on the acoustic energy decay model, the maximum likelihood estimation method may be applied for the energy based source localization problem. For convenience, all parameters refer to the same time window automatically. Define

$$\mathbf{Z} = \begin{bmatrix} \frac{y_1-\mu_1}{\sigma_1} & \frac{y_2-\mu_2}{\sigma_2} & \dots & \frac{y_N-\mu_N}{\sigma_N} \end{bmatrix}^T \tag{2.4}$$

Equation 2.3 can be simplified as:

$$\mathbf{Z} = \mathbf{GDS} + \xi \tag{2.5}$$

Where

$$\mathbf{S} = \begin{bmatrix} S_1 & S_2 & \dots & S_N \end{bmatrix}^T \tag{2.6}$$

$$\mathbf{S} = \operatorname{diag}\begin{bmatrix} \frac{g_1}{\sigma_1} & \frac{g_2}{\sigma_2} & \dots & \frac{g_N}{\sigma_N} \end{bmatrix} \tag{2.7}$$

$$\mathbf{D} = \begin{bmatrix} \frac{1}{d_{11}^2} & \frac{1}{d_{12}^2} & \cdots & \frac{1}{d_{1N}^2} \\ \frac{1}{d_{21}^2} & \frac{1}{d_{22}^2} & \cdots & \frac{1}{d_{2N}^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{d_{n1}^2} & \frac{1}{d_{n2}^2} & \cdots & \frac{1}{d_{NN}^2} \end{bmatrix} \tag{2.8}$$

d_{ij} the Euclidean distance between the i-th and j-th sensors. Then, $\xi_i = \frac{\varepsilon_i-\mu_i}{\sigma_i} \sim N(0,1)$, $\mathbf{Z}_i = \frac{y_i-\mu_i}{\sigma_i} \sim N(\frac{g_i}{\sigma_i}\sum_{j=1}^{N}\frac{S_j}{d_{ij}^2}, 1)$

Define

$$\theta = \begin{bmatrix} \mathbf{x}_1^T & \mathbf{x}_2^T & \dots & \mathbf{x}_N^T & S_1 & S_2 & \dots & S_N \end{bmatrix}^T \tag{2.9}$$

The log-likelihood function of equation (2.5) is:

$$l(\theta) = -\frac{1}{2}\|\mathbf{Z} - \mathbf{GDS}\|^2 \tag{2.10}$$

The maximum likelihood estimate of the parameters θ is the values that minimizes $l(\theta)$, or equivalently, maximizes

$$L(\theta) = \|\mathbf{Z} - \mathbf{GDS}\|^2 \tag{2.11}$$

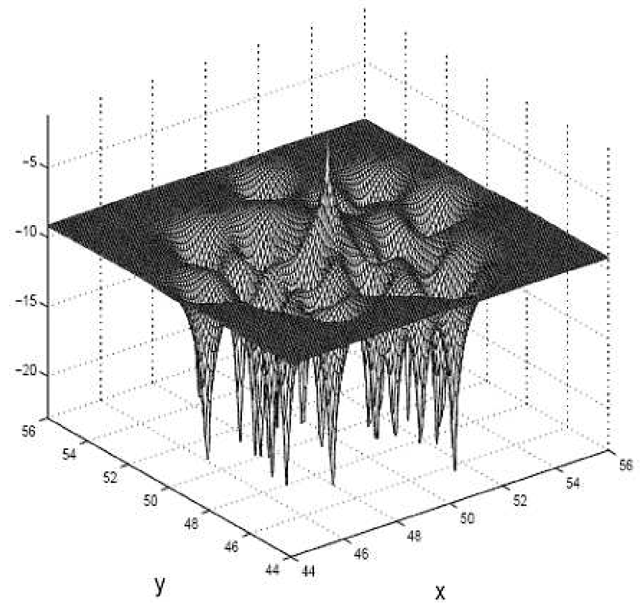

Figure 2.1: Negative log-likelihood Objective function

Figure 2.1 shows the negative log-likelihood function, which has multiple optima.

2.2.2 Distributed Gradient Search Approach

In the Centralized Maximum Likelihood estimation approach, the sensors have to transmit all of their data to a fusion center for processing. In large networks, the massive amount of energy and bandwidth required make such an approach impractical.

The driving philosophy behind the distributed approach is to balance estimator accuracy with the amount of communication required [1]. This is accomplished via in-network processing of data. An estimate of the source position is circulated through the network. Each node makes a small adjustment to the estimate based on its local data, and then passes the modified estimate to its neighbors. Similar to [22], assume the j-th measurement at sensor i takes the form $y_{ij} = s_i + \delta_{ij}$, where δ_{ij} are i.i.d samples of a white Gaussian noise process, and

$$s_i = \frac{A}{\|\theta - \mathbf{c}_i\|^{\beta}} \tag{2.12}$$

where A is the amplitude (energy) of the signal emitted by the source, and β is related to the attenuation characteristics of the medium through which the signal is begin transmitted. The maximum likelihood estimate for the location of a stationary source is

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{MN} \sum_{i=1}^{N} \sum_{j=1}^{M} (y_{ij} - \frac{A}{\|\theta - \mathbf{c}_i\|^{\beta}})^2 \tag{2.13}$$

where N is the number of sensors and M is the number of measurements at each sensor. Distributed gradient search approaches use a distributed incremental gradient algorithm to solve this non-linear least squares optimization iteratively.

2.2.2.1 Decentralized Incremental Optimization [1]

In this algorithm a parameter estimate is cycled through the network. Each node makes a small adjustment based on its local data when it receives the current estimate. Assume the sensors have been numbered $i = 1, 2, \ldots, N$ according to their order in the cycle. On the k-th cycle, sensor i receives an estimate $\psi_i^{(k)}$ from its neighbor and computes an update according to

$$\psi_i^{(k)} = \psi_{i-1}^{(k)} - \alpha \nabla f_i(\psi_{i-1}^{(k)}) \tag{2.14}$$

in which $\alpha > 0$ is the step size, $\nabla f_i(\psi_{i-1}^{(k)})$ denotes the gradient of f_i evaluated at $\psi_{i-1}^{(k)}$, and

$$f_i(\psi) = \sum_{j=1}^{M} (y_{ij} - \frac{A}{\|\psi - \mathbf{c}_i\|^{\beta}})^2 \tag{2.15}$$

The algorithm begins with an arbitrary initial condition $\psi_0^{(0)} = \hat{\theta}^{(0)}$, and after a complete cycle through the network one can get $\hat{\theta}^{(k)} = \psi_N^{(k)} = \psi_1^{(k+1)}$. The update is made by optimizing the function f_i which depends on local data at sensor i for the current estimate. The amount of in-network communication required is the number of cycles consequence. The tradeoff between accuracy and the amount of communication is quantified and can be controlled through the choice of a step size.

2.3 Localization Via Projection Onto Convex Sets

Traditional nonlinear least squares or maximum likelihood methods pose a difficult global optimization problem. As we can see in Figure 2.1, the objective function may have multiple local optima and saddle points and hence any local search method would can be trapped in suboptimal solution. [25] formulated the problem as a convex feasibility problem and applied the projection onto convex sets method in a distributed manner. Given the condition that the number of samples increase to infinity or in the absence of measurement noise, the convex feasibility problem has a unique solution at the true location when the source is located in certain area in the network.

2.3.1 Introduction of Projection Onto Convex Set Method

Consider a sensor network with N anchor nodes, whose positions are denoted as $\mathbf{c}_i \in \mathbb{R}^2$, $i = 1, \ldots, N$. Let $\mathbf{x} \in \mathbb{R}^2$ be the unknown coordinate of a non-anchor node, and let d_i be the measured distance between the node and the i-th anchor node. We assume exact distance measurements in the initial formulation, and absorb the influence of noisy distance measurements subsequently.

Following [22],[1], the reading of the source's signal strength at sensor i is modeled by

$$y_{ij} = \frac{A}{\|\psi - \mathbf{c}_i\|^{\beta}} + \delta_{ij} \tag{2.16}$$

Formulated as a nonlinear least squares problem, which is equivalent to maximum likelihood when the noise is Gaussian process, the parameter estimation is

$$\hat{\theta}_{NLS} = \arg\min_{\theta} \frac{1}{MN} \sum_{i=1}^{N} \sum_{j=1}^{M} (y_{ij} - \frac{A}{\|\theta - \mathbf{c}_i\|^{\beta}})^2 \tag{2.17}$$

However this objective function has multiple local optima, which the incremental gradient method may suffer from. As one can see in Figure 2.1, instead of a nonlinear least squares formulation 2.2.1 An alternative formulation of the problem is as follows. The function

$$f_i(\theta) = \sum_{i=1}^{N} (y_{ij} - \frac{A}{\|\theta - \mathbf{c}_i\|^{\beta}})^2 \tag{2.18}$$

obtains its minimum on the circle

$$\mathcal{C}_i = \left\{\theta \in \mathrm{I\!R}^2 : \|\theta - \mathbf{c}_i\| = d_i\right\} \tag{2.19}$$

where distance d_i is

$$d_i = \left[\frac{A}{\overline{y}_i}\right]^{1/\beta} \tag{2.20}$$

Let D_i be the ball defined by

$$D_i = \left\{\theta \in \mathrm{I\!R}^2 : \|\theta - \mathbf{c}_i\| \leq d_i\right\} \tag{2.21}$$

the estimation problem is solved by finding a point in the intersection of these balls.

$$\hat{\theta} \in D = \bigcap_{i=1}^{N} D_i \in \mathbb{R}^2 \tag{2.22}$$

Alternatively, the estimator would be any point that minimizes the sum of distances to the sets $D_i, i = 1, 2, \ldots, N$.

$$\hat{\theta} = \arg\min_{\theta \in \mathbb{R}^2} \sum_{i=1}^{N} \|\mathbf{x} - \mathcal{P}_{\mathbf{D_i}}(\mathbf{x})\| \tag{2.23}$$

where for a set $S \in \mathbb{R}^2$ and a point $\mathbf{x} \in \mathbb{R}^{\mathbf{2}}$, $\mathcal{P}_S(\mathbf{x})$is the orthogonal projection of $\mathbf{x}$ onto S given by

$$\mathcal{P}_S(\mathbf{x}) = \mathbf{arg} \min_{\mathbf{y} \in \mathbb{R}^{\mathbf{2}}} \|\mathbf{x} - \mathbf{y}\| \tag{2.24}$$

using the Euclidean norm $\| \cdot \|$. The convex hull, denoted by $\mathcal{H}$ is defined to be the smallest convex set containing the anchor nodes:

$$\mathcal{H} = \left\{ \mathbf{y} \in \mathbb{R}^2 : \mathbf{y} = \sum_{i=1}^{N} \alpha_i \mathbf{c}_i,\ \alpha_i \geq 0,\ \sum_{i=1}^{N} \alpha_i = 1 \right\} \tag{2.25}$$

Figure 2.2 shows geometrical shape of the convex hull of the anchor nodes. the convex feasibility problem (2.23) has a unique solution at the true source's location only when the true location is inside the convex hull $\mathcal{H}$ [25].

Figure 2.3 illustrates that when the source to be located is inside $\mathcal{H}$, intersection of convex sets D_i is a unique point, and thus the convex feasibility problem (2.23) has a unique solution. However, Figure 2.4 shows that when the source is outside the convex hull, the intersection of convex sets D_i is an area, and thus convex feasibility problem

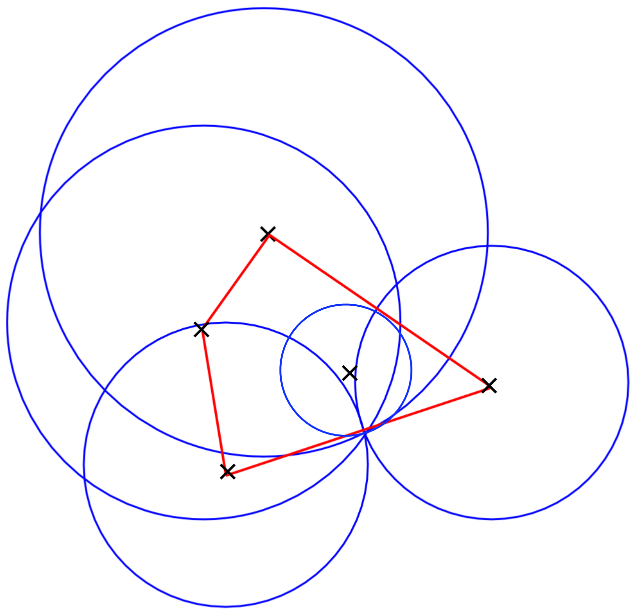

Figure 2.2: Convex hull of the anchor nodes.

(2.23) has no unique solution. A formulation which overcomes this shortcoming is developed in section ~refsec:Hpocs

2.3.1.1 Hyperbolic Projection Onto Convex Sets Algorithm

To overcome the shortcoming of [25], the author of [27] proposed a hyperbolic POCS algorithm, an augmentation to the POCS method which successfully converges (in the absence of noise) to the correct position estimate for sensors lying outside the convex hull $\mathcal{H}$. A performance penalty is observed, however, using the hyperbolic POCS algorithm on sensors inside the convex hull, thus motivating a hybrid algorithm [27] that aims to leverage the strengths of the hyperbolic algorithm and its circular predecessor [25]. We illustrate next section how the shortcomings of these earlier

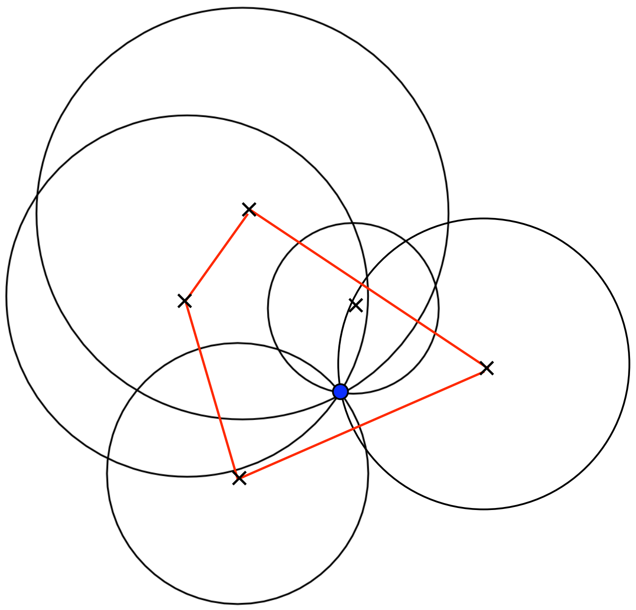

Figure 2.3: Sensor lies inside convex hull:unique solution.

algorithms can be overcome by projecting instead onto the boundaries of specific convex sets.

2.4 Other Sensor Network Localization Algorithms

The POCS algorithm is very effective in overcoming the local convergence problem when the source is located within the convex hull of the sensor nodes. However, the POCS algorithm does suffer from poor performance when the source resides outside the convex hull. There are many other approach proposed to solved the localization problem in wireless sensor networks.

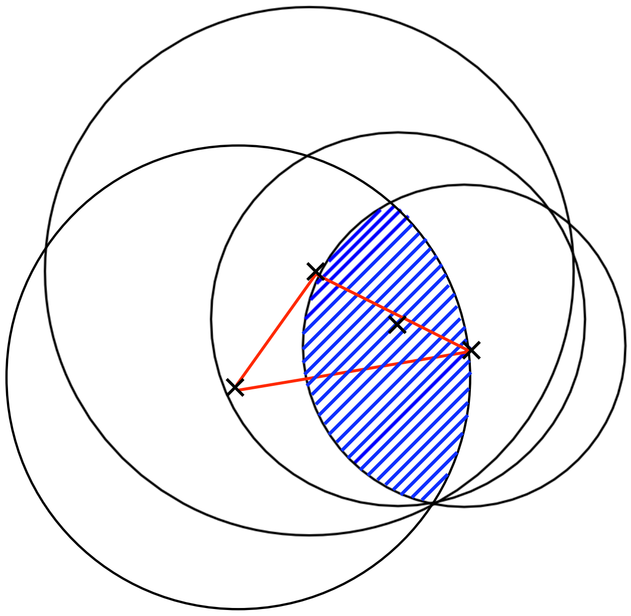

Figure 2.4: Sensor lies outside convex hull:no unique solution.

2.4.1 Semidefinite Programming (SDP) Approach

By transforming the source localization problem into a convex optimization problem using minimax approximation and semidefinite relaxation, one can find the global minimum of the relaxed problem, with no need to impose coverage conditions [28]. The major impediment of this approach, however, is the rather high computational complexity.

2.4.2 Multidimensional Scaling (MDS) Algorithms

Multidimensional scaling (MDS) algorithms formulate sensor localization from range measurements as a least square problem [29]. In classical MDS, the LS solution is found by an eigenvalue decomposition, which does not suffer from local maxima. To linearize the localization problem, the classical MDS formulation works with squared distance rather than distance itself, and the end result is very sensitive to range measurement errors [3]. Other MDS-based techniques, not based on eigenvalue-decompositions, can be made more robust by allowing measurements to be weighted according to their perceived accuracy [30].

2.5 Summary

As we have reviewed in this chapter, traditional localization methods encounter the multimodality problem that plagues least-squares formulations. Projections Onto Convex Sets algorithm formulats the problem as a convex feasibility problem that overcomes previous shortcomings. However, the solution of Projection Onto Convex Set algorithm might not be unique. This is another impetus to develop a new approach, based on the projections Onto Boundary Sets. Our new Algorithm is introduced in detail in the next chapter.

Chapter 3

New Localization Algorithm: Projection Onto the Boundary Sets

3.1 Introduction

Our new algorithm considers localization of nodes in a sensor network using distance measurements. Previous method of projection onto convex sets overcomes the multimodality problem that plagues least-squares formulations. Previous efforts in this direction require either that the sensor be located in the convex hull of the anchor nodes, or that complicated hyperbolic geometric calculations be employed. Here we propose a new algorithm which projects onto the boundary of convex sets, and features a computationally simple update procedure. Both cyclic and random projection schedules are considered, and initial convergence proofs are offered.

3.2 Background

As introduced in previous chapters, we consider a sensor network with N anchor nodes, whose positions are denoted as $\mathbf{c}_i \in \mathbb{R}^2$, $i = 1, \ldots, N$. Let $\mathbf{x} \in \mathbb{R}^2$ be the unknown coordinate of a non-anchor node, and let d_i be the measured distance between the node and the i-th anchor node. We assume exact distance measurements in the initial formulation, and absorb the influence of noisy distance measurements subsequently.

3.3 New Algorithm: Projection onto Boundary Sets

With d_i denoting the distance between the true location $\mathbf{x}$ and the anchor node $\mathbf{c}_i$, the boundary set $\partial\mathcal{C}_i$ is given as

$$\partial\mathcal{C}_i = \left\{\mathbf{y} \in \mathbb{R}^2 : \|\mathbf{y} - \mathbf{c}_i\| = d_i\right\} \tag{3.1}$$

using the Euclidean norm $\|\cdot\|$. If the true distances d_i are used, then clearly the point $\mathbf{x}$ lies at the intersection of the sets $\{\partial\mathcal{C}_i\}$:

$$\mathbf{x} \in \bigcap_{i=1}^{N} \partial\mathcal{C}_i \tag{3.2}$$

Moreover, save for degenerate situations (e.g., anchor nodes all on a line), the true position is the unique intersecting point of the boundary sets.

Now, the projection of an arbitrary point $\mathbf{y} \in \mathbb{R}^2$ onto the boundary set $\partial\mathcal{C}_i$, denoted $\mathcal{P}_i(\mathbf{y})$, is readily calculated as

$$\mathcal{P}_i(\mathbf{y}) = \mathbf{c}_i + d_i \, \frac{\mathbf{y} - \mathbf{c}_i}{\|\mathbf{y} - \mathbf{c}_i\|} \tag{3.3}$$

and satisfies

$$\mathcal{P}_i(\mathbf{y}) = \arg \min_{\mathbf{z} \in \partial\mathcal{C}_i} \|\mathbf{y} - \mathbf{z}\| \,. \tag{3.4}$$

This motivates a cyclic projection algorithm that seeks the intersection of the boundaries, given as follows:

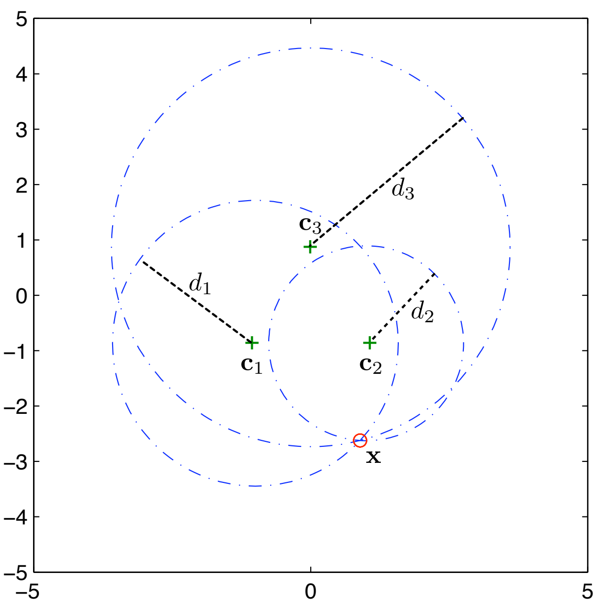

Figure 3.1: Anchor nodes indicated by “+”, with true location as “○”.

1. Initialization: $\mathbf{x}^0$ is arbitrary.

2. Iterative step: for $k = 1, 2, \ldots,$

$$\mathbf{x}^k = \mathcal{P}_{k \bmod N}(\mathbf{x}^{k-1}) \tag{3.5}$$

   in which $k \mod N$ cycles through the set $\{1, 2, \ldots, N\}$.

Figure 3.1 shows three anchor nodes (indicated as “+”) along with the circles which collect points $\partial\mathcal{C}_i$ at the true distance d_i; the true location (“○”) lies at the intersection $\cap_i \partial\mathcal{C}_i$. Figure 3.2 shows a typical run, illustrating convergence of the position estimates to the true location, even though the true location lies outside the convex hull $\mathcal{H}$, thus overcoming the shortcoming of [25] while remaining computationally simpler than the hyperbolic variant from [27].

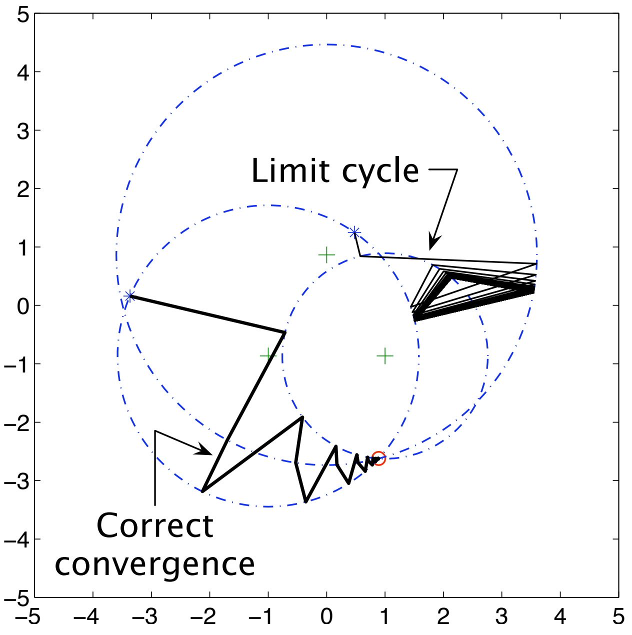

Figure 3.2: Illustrating convergence versus a limit cycle, depending on the initialization.

An observed weakness of the proposed algorithm, however, is its susceptibility to limit cycles in some cases. Figure 3.2 likewise illustrates one such case, in which successive iterates $\mathbf{x}^k$ become trapped in a triangle. Such limit cycles have been observed for arbitrary numbers of anchor nodes for specific alignments (such as the equilateral triangle illustrated here, when using three anchor nodes). A slight repositioning of the anchor nodes may be observed to break the limit cycle, indicating that, in a probabilistic sense, the likelihood of encountering one may be small. The example of figure 3.2, however, indicates that this likelihood remains nonzero.

We have observed in simulations that if we alter the sequence of projection steps, limit cycles can be broken. This suggests that a more effective way to eliminate such

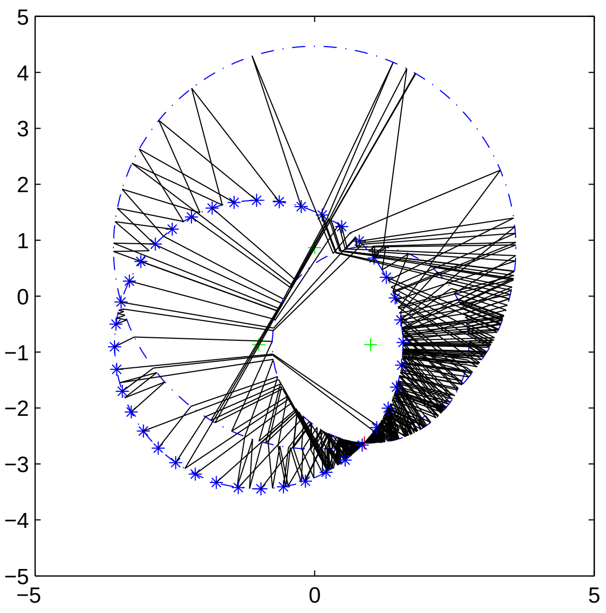

Figure 3.3: Convergence for various initial points, using the random projection schedule. Each converges to the true location.

limit cycles is to use a random (rather than cyclic) projection algorithm, summarized as follows:

- Initialization: $\mathbf{x}^0$ is arbitrary, and set $j_1 = 1$.

- Iterations: for $k = 1,\ 2,\ \ldots,$

$$\mathbf{x}^k = \mathcal{P}_{j_k}(\mathbf{x}^{k-1}) \tag{3.6}$$

$$j_{k+1} = \text{rand}\left(\{1, 2, \ldots, N\}\backslash j_k\right) \tag{3.7}$$

in which rand$(\cdot)$ returns an element chosen at random from its argument list.

Figure 3.3 shows the same situation as figure 3.2, using now a random projection sequence with many different initializations; each run is observed to convergence to

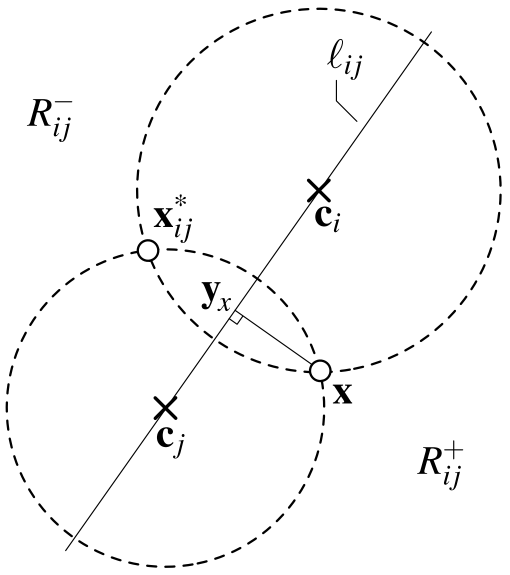

Figure 3.4: Illustrating half-planes R_{ij}^+ and R_{ij}^-.

the true location.

3.4 Convergence Aspects

For coherence we consider first the case of two anchor nodes, and then extend the results to an arbitrary number of anchor nodes.

Figure 3.4 illustrates that, with two anchor nodes, the intersection $\partial\mathcal{C}_i \cap \partial\mathcal{C}_j$ gives two solutions: the true point $\mathbf{x}$ and its "conjugate" $\mathbf{x}_{ij}^*$. The half-plane of points closer to $\mathbf{x}$ than to its conjugate $\mathbf{x}_{ij}^*$ is denoted R_{ij}^+, and is bounded by the line passing through $\mathbf{c}_i$ and $\mathbf{c}_j$.

We introduce a discrepancy function

$$D_{ij}(\hat{\mathbf{x}}) = \frac{\|\hat{\mathbf{x}} - \mathbf{x}\|}{\|\hat{\mathbf{x}} - \mathbf{x}^*_{ij}\|} \tag{3.8}$$

which is nonnegative and vanishes only at $\hat{\mathbf{x}} = \mathbf{x}$. It is easy to check that $D_{ij}(\hat{\mathbf{x}})$ is bounded in R^+_{ij}:

$$D_{ij}(\hat{\mathbf{x}}) < 1 \qquad \Leftrightarrow \qquad \hat{\mathbf{x}} \in R^+_{ij}$$

Note that, with only two anchor nodes, the random projection algorithm effectively reduces to the periodic projection algorithm.

Lemma 3.1 *Projections between circles centered at $\mathbf{c}_i$ and $\mathbf{c}_j$ preserve half-plane membership: $\hat{\mathbf{x}}^k \in R^+_{ij}$ if and only if $\hat{\mathbf{x}}^{k-1} \in R^+_{ij}$.*

A proof is offered in Appendix A.

Theorem 3.1 *Consider the update equations that alternately project onto circles centered at $\mathbf{c}_i$ and $\mathbf{c}_j$.*

1. *Given any initialization $\hat{\mathbf{x}}^0$ for which $D_{ij}(\hat{\mathbf{x}}^0) < 1$, convergence is monotonic:*

$$D_{ij}(\hat{\mathbf{x}}^k) < D_{ij}(\hat{\mathbf{x}}^{k-1}), \qquad \textit{for all } \hat{\mathbf{x}}^k \neq \hat{\mathbf{x}}^{k-1};$$

2. *The local convergence rate is linear: for $D_{ij}(\hat{\mathbf{x}}^{k-1})$ sufficiently small,*

$$D_{ij}(\hat{\mathbf{x}}^k) = D_{ij}(\hat{\mathbf{x}}^{k-1}) \cos\theta_{ij} \tag{3.9}$$

where θ_{ij} is the subspace angle between $\mathbf{x} - \mathbf{c}_i$ and $\mathbf{x} - \mathbf{c}_j$.

For the proof, suppose $\hat{\mathbf{x}}^0$ is in the same half-plane as $\mathbf{x}$ (i.e., $\hat{\mathbf{x}}^0 \in R_{ij}^+$); by Lemma 1 all iterates $\hat{\mathbf{x}}^k$ remain in R_{ij}^+ as well. Consider the unit-norm vector

$$\mathbf{u} = \frac{\hat{\mathbf{x}}^{k-1} - \mathbf{c}_j}{\|\hat{\mathbf{x}}^{k-1} - \mathbf{c}_j\|}\,, \tag{3.10}$$

which relates to successive iterates as

$$\hat{\mathbf{x}}^{k-1} = \mathbf{c}_j + r\,\mathbf{u} \tag{3.11}$$

$$\hat{\mathbf{x}}^{k} = \mathbf{c}_j + d_j\,\mathbf{u} \tag{3.12}$$

with $r = \|\hat{\mathbf{x}}^{k-1} - \mathbf{c}_j\|$. The line $\mathbf{c}_j + \rho\,\mathbf{u}$, parameterized by a nonnegative scalar ρ, then contains the iterates $\hat{\mathbf{x}}^{k-1}$ and $\hat{\mathbf{x}}^k$ at $\rho = r$ and $\rho = d_j$, respectively. The inequality $D_{ij}(\hat{\mathbf{x}}^k) < D_{ij}(\hat{\mathbf{x}}^{k-1})$ will follow upon showing that

$$d_j = \arg\min_{\rho} D_{ij}(\mathbf{c}_j + \rho\,\mathbf{u}). \tag{3.13}$$

To this end, a straightforward calculation shows that

$$\frac{\partial}{\partial \rho}[D_{ij}(\mathbf{c}_j + \rho\,\mathbf{u})]^2 = (\rho^2 - d_j^2)\,\frac{2\,\langle \mathbf{u}, \mathbf{x} - \mathbf{x}^* \rangle}{\|\mathbf{c}_j + \rho\,\mathbf{u} - \mathbf{x}^*\|^4} \tag{3.14}$$

whose sole zero in $\rho > 0$ lies at $\rho = d_j$. As $D(\hat{\mathbf{x}}) \leq 1$ in R_{ij}^+, we have in particular that $D(\mathbf{c}_j + \rho\,\mathbf{u}) \leq 1$ for all $\rho \geq 0$. Indeed, as the maximum $D_{ij}(\mathbf{c}_j + \rho\,\mathbf{u}) = 1$ is attained only at $\rho = 0$ and in the limit as $\rho \to \infty$, the critical point $\rho = d_j$ must be a minimum.

Interchanging the role of i and j shows that $\hat{\mathbf{x}}^{k+1}$, lying on the circle centered at $\mathbf{c}_i$, fulfills $D_{ij}(\hat{\mathbf{x}}^{k+1}) < D_{ij}(\hat{\mathbf{x}}^k)$ as well, to establish the first part of the theorem.

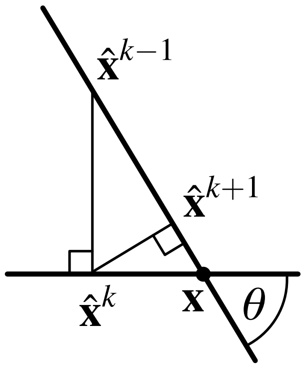

Figure 3.5: Zooming in near the intersection of two circles.

For the second part, we may zoom in near the intersection $\mathbf{x}$; locally the two circles may be replaced by their tangent lines, as in figure 3.5. As each projection gives a displacement normal to a tangent line, we see by inspection that

$$\|\hat{\mathbf{x}}^k - \mathbf{x}\| = \|\hat{\mathbf{x}}^{k-1} - \mathbf{x}\| \cos\theta_{ij}. \tag{3.15}$$

Moreover, points sufficiently close to $\mathbf{x}$ are nearly equidistant from $\mathbf{x}^*_{ij}$: $\|\hat{\mathbf{x}}^k - \mathbf{x}^*_{ij}\| \approx \|\hat{\mathbf{x}}^{k-1} - \mathbf{x}^*_{ij}\|$. This approximation becomes increasingly accurate as $\mathbf{x}$ is approached, so that dividing both sides of the equality (3.15) by $\|\hat{\mathbf{x}}^k - \mathbf{x}^*_{ij}\|$ gives the second part of the theorem. $\diamond$

When projecting between more than two circles, the local convergence properties follow similarly to theorem 1: if the next projection $\hat{\mathbf{x}}^{k+1}$ goes to the circle centered at $\mathbf{c}_\ell$, then by the same geometric consideration of Figure 3.5 we have

$$\|\hat{\mathbf{x}}^{k+1} - \mathbf{x}\| = \|\hat{\mathbf{x}}^k - \mathbf{x}\| \cos\theta_{j\ell}. \tag{3.16}$$

Thus the convergence rate is asymptotically limited by the poorest angular separation

$\max_{i,j} \cos\theta_{ij}$ of anchor nodes, as seen from the true location.

The "global" convergence (i.e., from points too far from $\mathbf{x}$ to allow the circles to be approximated by tangent lines) is more delicate when using more than two anchor nodes. One approach to studying convergence is to average, in some sense, the various pairwise discrepancy functions $D_{ij}(\hat{\mathbf{x}})$. With N anchor nodes, we have $M = \binom{N}{2} = N(N-1)/2$ such discrepancy functions, and different average choices include:

$$\begin{aligned} F_1(\hat{\mathbf{x}}) &= \frac{1}{M}\sum_{i,j} D_{ij}(\hat{\mathbf{x}}) \qquad \text{(arithmetic mean)} \qquad (3.17)\\ F_2(\hat{\mathbf{x}}) &= \left(\prod_{i,j} D_{ij}(\hat{\mathbf{x}})\right)^{1/M} \qquad \text{(geometric mean)} \qquad (3.18)\\ F_3(\hat{\mathbf{x}}) &= \frac{M}{\sum_{i,j}[1/D_{ij}(\hat{\mathbf{x}})]} \qquad \text{(harmonic mean)} \qquad (3.19) \end{aligned}$$

It is straightforward to check that

$$F_1(\hat{\mathbf{x}}) \geq F_2(\hat{\mathbf{x}}) \geq F_3(\hat{\mathbf{x}}), \qquad \text{for all } \hat{\mathbf{x}}.$$

Indeed, $F_1(\hat{\mathbf{x}}) \geq F_2(\hat{\mathbf{x}})$ follows from the arithmetic-mean–geometic-mean inequality, as does $F_2(\hat{\mathbf{x}}) \geq F_3(\hat{\mathbf{x}})$ upon observing that

$$\frac{F_3(\hat{\mathbf{x}})}{F_2(\hat{\mathbf{x}})} = \frac{\left(\prod_{i,j} \|\hat{\mathbf{x}} - \mathbf{x}^*_{ij}\|\right)^{1/M}}{\frac{1}{M}\sum_{i,j} \|\hat{\mathbf{x}} - \mathbf{x}^*_{ij}\|} \qquad (3.20)$$

We have observed in simulations that, when using more than two anchor nodes, monotonic convergence kicks in once $F_i(\hat{\mathbf{x}}) < 1$, although a proof in the general case is still lacking.

3.5 Initialization

A well-chosen initialization point $\hat{\mathbf{x}}^0$ will generally lead to surer convergence to the correct position estimate. Two cases may be considered:

- **Case 1:** The true location $\mathbf{x}$ lies in the convex hull of the anchor nodes ($\mathbf{x} \in \mathcal{H}$). In this case, convergence to the correct position estimate follows by arguments similar to those of [6].

- **Case 2:** The true location $\mathbf{x}$ lies outside the convex hull of the anchor node ($\mathbf{x} \notin \mathcal{H}$).

For case 2, an effective initialization strategy consists in first sorting the available distance measurements to select the smallest; let this index be denoted n. Then choose $\hat{\mathbf{x}}^0$ in a vicinity of $\mathbf{c}_n$. Let i and j be the indices of two of the larger distances, and project back and forth between circles centered at $\mathbf{c}_i$ and $\mathbf{c}_j$ for a few iterations, and then switch to the random projection strategy of section 3.3. In essence, when the true location lies outside $\mathcal{H}$, this strategy gives a high probability of the initial position estimates lying in R_{ij}^+, as desired in view of Theorem 1. The subsequent switch to the random projection scheme then relies on the local convergence of the algorithm.

This same initialization strategy also appears effective for case 1, as the true position $\mathbf{x}$ is already in $\mathcal{H}$ anyway.

3.6 Noisy Measurements

With random noise contaminating the distance measurements $\{d_i\}$, the position estimates become random variables. Let

$$d_i = \overline{d}_i + \delta_i \tag{3.21}$$

in which $\overline{d}_i$ is the true distance to the i-th anchor, and δ_i is the measurement error. The influence of δ_i is to move the boundary of each circle, so that projections will land systematically inside (resp., outside) the circle $\partial\mathcal{C}_i$ if $\delta_i < 0$ (resp., $\delta_i > 0$). For small displacements, such that circular arcs can be replaced by their tangent lines at the point of intersection, the actual intersection between circles $\partial\mathcal{C}_i$ and $\partial\mathcal{C}_j$ is displaced to the opposite corner of a parallelogram, as depicted in figure 3.6. From basic trigonometry, the distance Δ_{ij} between the true location and the displaced intersection is

$$\Delta_{ij} = \frac{\delta_i^2 + 2\delta_i\delta_j\cos\theta_{ij} + \delta_j^2}{\sin\theta_{ij}} \tag{3.22}$$

involving the subspace angle θ_{ij} between the anchored displacements $(\mathbf{x} - \mathbf{c}_i)$ and $(\mathbf{x} - \mathbf{c}_j)$. With N anchor nodes, we have $N(N-1)/2$ displaced intersections between circles, which now function as competing local attractors to the position estimator. The algorithm will then converge towards a vicinity of these competing attractors, and then "rattle around" [31], with the worst-case deviation from the true location given by $\max_{ij}\Delta_{ij}$.

Figure 3.7 shows a simulation run in which the distance measurements are con-

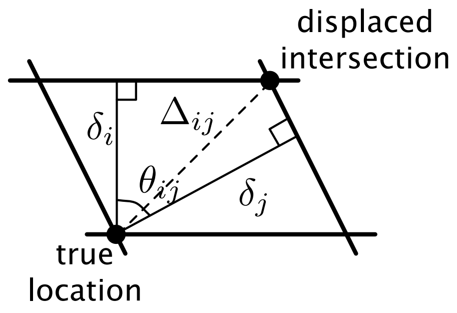

Figure 3.6: Illustrating displaced intersection of circles $\partial\mathcal{C}_i$ and $\partial\mathcal{C}_j$ in the presence of small distance errors δ_i and δ_j.

taminated with errors, with $\sigma_i^2 = 0.04$. The position estimates are seen to converge towards the true location, and then "rattle around". By averaging the position estimates over many such runs, the mean position was observed to be unbiased, with a variance given by σ_i^2.

3.7 Concluding Remarks

The modified projection algorithm proposed here, using projections to the boundaries of sets, is observed to perform well even when the true position lies outside the convex hull of the anchor nodes. The algorithm also features a simple update law and behaves benignly in the presence of measurement errors.

Although we have focused on the two-dimensional case, the algorithm extends readily to three-dimensional data as well; figure 3.8 gives an example run.

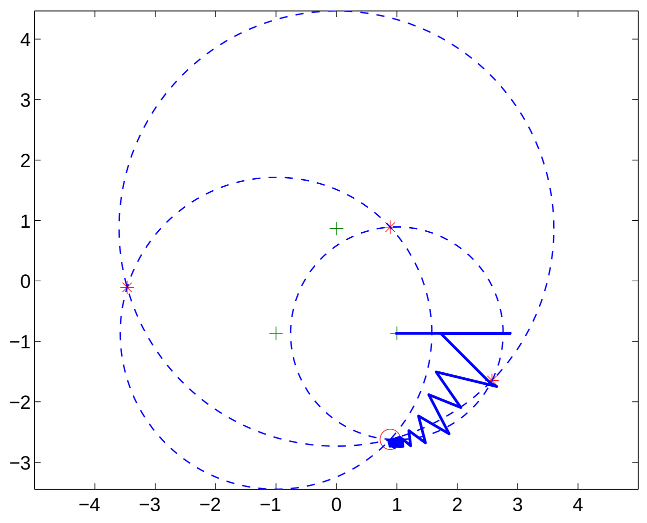

Figure 3.7: Position estimates using noisy distance measurements.

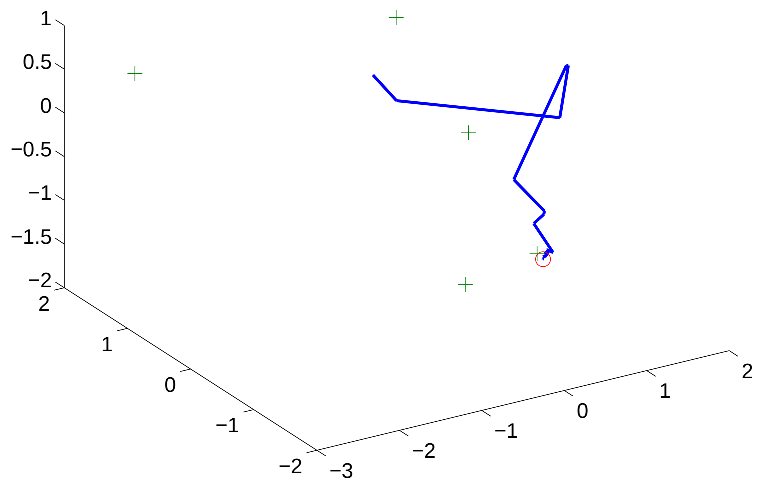

Figure 3.8: Illustrating convergence for three-dimensional data.

Chapter 4

Distance Information Reconstruction For Wireless Sensor Network Localization

4.1 Introduction

Most of localization algorithms are based on the critical information–distance measurements. However, in practice, distance information may not be complete, as some distance measurements might not be available, others might be contaminated by noise. In this chapter, we focus on the distance information reconstruction problem.

Specific distance measurements can be obscured due to various factors, including bursty interference or transient jamming during distance measurements, or inadvertent obstacles impeding communication between select sensor pairs. In such cases, the various localization methods may have insufficient input data, necessitating estimation of the missing distance measurements from those available. The distance reconstruction problem has considerable geometric structure [15], [16] and we examine further in this work the structural features exploited in [16] based on the low-rank character of a certain matrix constructed from the pairwise distances.

While convex programming can provide powerful algorithmic solutions to the low-rank matrix reconstruction problems, existing approaches appear ill suited to integrating inertial constraints which underlie Euclidean distance matrices [32], [33], suggesting that alternative reconstruction algorithms might better capture the problem structure if the inertial constraints can somehow be absorbed: inertia is more informative than rank, since rank can be inferred from inertia, but not vice-versa. The intent of this chapter is to propose such inertial constrained iterative algorithms for distance matrix reconstruction, and offer some initial performance comparisons illustrating their interest.

Section 4.2 reviews some basic results of multidimensional scaling, in order to recall specific results on the inertia of distance matrices that serve as consistency checks in the iterative algorithms developed in this chapter. The issue of unique reconstruction of missing distances from those remaining is reviewed in Section 4.3, to ensure meaningful simulation examples are constructed. Further algebraic structure of the reconstruction problem is developed in Section 4.4, which serves to illustrate how nonuniqueness can plague solutions that exploit only rank; further inconsistencies of existing algorithms are identified in Section 4.5, thus motivating the new algorithms proposed in Section 4.6. Simulation examples are included in Section 4.7, with concluding remarks sythesized in Section 4.8.

4.2 Problem Structure

4.2.1 Multidimensional Scaling

We review in this section results from multidimensional scaling relating to consistent distance measurements between sensor nodes.

An $N \times N$ symmetric matrix $\mathbf{D}$ is a *pre-Euclidean distance matrix* [32], [33], [34] if it has zeros on the main diagonal and nonnegative elements elsewhere. If in addition there exist points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ in $\Re^m$ such that

$$\mathbf{D}_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \tag{4.1}$$

then $\mathbf{D}$ becomes a Euclidean distance matrix, and m is the dimension of the space containing the points $\mathbf{x}_1, \ldots, \mathbf{x}_N$.

We will be concerned principally with the case $m = 2$ here, although most of the results extend readily to higher m as well, with the important exception of Theorem 4.1 below.

For the context at hand, consider a network of N sensors randomly distributed in a plane, with $\mathbf{x}_i \in \Re^2$ the position of the i-th sensor. The squared distance between sensor pairs becomes

$$d_{ij}^2 \triangleq \|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{x}_i^T \mathbf{x}_i + \mathbf{x}_j^T \mathbf{x}_j - 2\,\mathbf{x}_i^T \mathbf{x}_j\,. \tag{4.2}$$

We collect the sensor coordinates into the matrix

$$\mathbf{X} = [\mathbf{x}_1 \;\; \mathbf{x}_2 \;\; \cdots \;\; \mathbf{x}_N]^T \in \Re^{N \times 2}. \tag{4.3}$$

The Euclidean distance matrix $\mathbf{D} \in \Re^{N\times N}$ for these points becomes [35], [17], [16]

$$\mathbf{D} = \mathbf{y}\,\mathbf{1}^T + \mathbf{1}\,\mathbf{y}^T - 2\,\mathbf{X}\mathbf{X}^T, \tag{4.4}$$

in which

$$\mathbf{y}^T = \left[\|\mathbf{x}_1\|^2 \;\; \|\mathbf{x}_2\|^2 \;\; \cdots \;\; \|\mathbf{x}_N\|^2\right] \in \Re^{1\times N} \tag{4.5}$$

and $\mathbf{1}$ denotes the $N \times 1$ vector of all ones. This shows that $\mathbf{D}$ has rank at most four [35], [16].[1] Note that $\mathbf{D}$ is invariant to rotation, translation, and reflection; resolving these indeterminacies requires that specific nodes have known locations.

Introducing the matrix $\mathbf{B} \in \Re^{(N-1)\times(N-1)}$ with entries

$$\mathbf{B}_{ij} = d_{iN}^2 + d_{jN}^2 - d_{ij}^2\,, \qquad 1 \le i, j \le N{-}1, \tag{4.6}$$

one shows [35] that the distances $\{d_{ij}\}$ are compatible with N points in $\Re^2$ if and only if $\mathbf{B}$ is positive semi-definite, having rank 2. Indeed, in that case, $\mathbf{B}$ factors as

$$\mathbf{B} = 2\,\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T, \tag{4.7}$$

where $\widetilde{\mathbf{X}} \in \Re^{(N-1)\times 2}$ has as its i-th row $(\mathbf{x}_i - \mathbf{x}_N)^T$.

A similar factorization is exploited in isomap [36]–[38], using row and column centering. Specifically, introduce the projection matrix

$$\mathbf{P} = \mathbf{I}_N - \frac{\mathbf{1}\mathbf{1}^T}{N} \tag{4.8}$$

This is a centering matrix, in the sense that if $\mathbf{w} = \mathbf{P}\mathbf{v}$, their components relate as $w_i = v_i - (1/N)\sum_j v_j$, which is seen to subtract the mean $(1/N)\sum_j v_j$ from each

[1]The formulation of [35] actually considers an augmented version of $\mathbf{D}$, but reaches the same rank conclusion.

component. Since clearly $\mathbf{P1} = \mathbf{0}$, centering both rows and columns of $\mathbf{D}$ yields again a rank two matrix:

$$\begin{aligned} \mathbf{PDP} &= \mathbf{P}\left(\mathbf{y}\,\mathbf{1}^T + \mathbf{1}\,\mathbf{y}^T - 2\,\mathbf{X}\mathbf{X}^T\right)\mathbf{P} \\ &= -2(\mathbf{PX})(\mathbf{PX})^T \end{aligned} \tag{4.9}$$

This has essentially the same structure as (4.6) above: aside from a sign change and difference in size, the coordinates of (4.9) are oriented around their center of mass, whereas (4.6) has them "centered" with respect to the final node $\mathbf{x}_N$. The form (4.6) will prove more convenient in subsequent algorithm design in section 4.6.

4.2.2 Inertial Constraints

If $\mathbf{v}$ is any vector orthogonal to $\mathbf{1}$, then $\mathbf{v}^T\mathbf{D}\mathbf{v} \leq 0$ [32], [33], i.e., a Euclidean distance matrix is negative semi-definite in the subspace of zero mean vectors. This basic relation has implications on the inertia of $\mathbf{D}$, which we review here as they play an important role in later consistency checks.

We recall that the inertia of a symmetric matrix [39], [40] is the couple (ρ_+, ρ_-) where ρ_+ (resp., ρ_-) is the number of positive (resp., negative) eigenvalues of the matrix; the rank is therefore $\rho = \rho_+ + \rho_-$.

Property 4.1 *The Euclidean distance matrix* $\mathbf{D}$ *has one eigenvalue positive* ($\rho_+ = 1$)*, with all other nonzero eigenvalues negative.*

Thus if $\mathbf{D}$ has rank four, its inertia is $(1, 3)$. Geometries giving rank less than four are noted in the remark below.

Proof: Note that $\mathbf{D}$ can be written as

$$\mathbf{D} = \begin{bmatrix} \mathbf{1} & \mathbf{y} & \mathbf{X} \end{bmatrix} \underbrace{\begin{bmatrix} & 1 & \\ 1 & & \\ & & -2\,\mathbf{I}_2 \end{bmatrix}}_{\mathbf{C}} \begin{bmatrix} \mathbf{1}^T \\ \mathbf{y}^T \\ \mathbf{X}^T \end{bmatrix}, \tag{4.10}$$

where $\mathbf{C}$ has eigenvalues at $+1$, -1, -2 and -2, and thus inertia $(1,3)$. By a simple extension of Sylvester's law of inertia [39] (which asserts that inertia is invariant to a congruence transformation), it follows that $\mathbf{D}$ can have at most one eigenvalue positive: $\rho_+ \leq 1$. To show indeed $\rho_+ = 1$, it suffices to find a nonzero $\mathbf{v} \in \Re^N$ for which $\mathbf{v}^T\mathbf{D}\mathbf{v} > 0$. Choosing $\mathbf{v} = \mathbf{1}$, we have

$$\mathbf{v}^T\mathbf{D}\mathbf{v} = \sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}^2 > 0. \qquad \diamond$$

Remark: The factor $\begin{bmatrix} \mathbf{1} & \mathbf{y} & \mathbf{X} \end{bmatrix}$ can drop below rank 4 when $y_i = \|\mathbf{x}_i\|^2$ is the same for each index i, placing the sensors in a circle. For this case, the rank of $\mathbf{D}$ can be no larger than 3, having inertia $(1,2)$. Similarly, the term $\mathbf{X}$ drops from rank 2 to rank 1 when all sensors are on a line, giving again $\mathbf{D}$ having rank at most 3. For other sensor geometries, the matrix $\mathbf{D}$ will have rank 4 with high probability, giving inertia $(1,3)$. $\diamond$

4.2.3 Basic Problem Statement

Let Ω denote the set of index pairs (i, j) for which d_{ij} is known; its complement $\overline{\Omega}$ thus contains index pairs (i, j) for which d_{ij} must be estimated.

Problem 4.1 *Given a partial description of the Euclidean distance matrix*

$$\widehat{\mathbf{D}}_{ij} = \begin{cases} d_{ij}^2, & (i,j) \in \Omega; \\ \\ *, & (i,j) \in \overline{\Omega}; \end{cases} \tag{4.11}$$

with "$*$" a missing entry, find the missing entries, i.e., reconstruct $\mathbf{D}$.

A variant on this formulation, in which distance measurements are known with confidence values, is briefly addressed in Section 4.5.2

4.3 Minimality and Uniqueness

Before developing reconstruction algorithms, we treat the question of the minimal number of distance measurements necessary to reconstruct those remaining, and whether a unique solution is consistent with the given distance measurements. We suppose that the available distance measurements are exact in this section.

Given N sensors, we clearly have $2N$ parameters to specify their locations in the plane. Their pairwise distances, however, are insensitive to translation (two degrees of freedom) and rotation (an additional degree of freedom) in the plane, reducing the number of degrees of freedom to $2N - 3$; fewer than $2N - 3$ distance measurements are thus insufficient to reconstruct those remaining.

This "minimal" number $2N-3$, however, still proves insufficient, as we review below. ($2N-2$ turns out to be the minimal number, but is sufficient only for specific sensor geometries.) Necessary and sufficient conditions for uniqueness are developed in the insightful work of Aspnes *et al.* [15], whose basic results are briefly reviewed here, for the purpose of ensuring meaningful simulation scenarios in section 4.7.

Let the nodes $\{\mathbf{x}_i\}_{i=1}^N$ be collected into a vertex set $\mathcal{V}$ of a graph, and let the set of edges $\mathcal{E}$ collect the known distance pairs $\{d_{ij}\}$, with edge lengths commensurate with these distances, as in Figure 4.1. A graph is *first order rigid* if the given distances fix the nodes locally with respect to each other, i.e., there is no local displacement of the nodes consistent with the given distance values, aside from the trivial indeterminacies of translation and rotation of all nodes together. (Rigidity is a common notion in mechanical and civil engineering: One may think of the edges as steel bars of prescribed lengths $\{d_{ij}\}$, whose endpoints are numbered according to the node pair (i,j) they connect; rigidity means that once the steel bars are connected, the nodes are "locked" into position by virtue of their geometry). Clearly, if the graph is not rigid, the remaining distances cannot be uniquely determined.

Numerous means of testing first-order rigidity are available [15]; we content ourselves here with the rank of a "rigidity matrix" [41], [42]. Specifically, with $d_{ij}^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2$, all local displacements $\Delta\mathbf{x}_i$ and $\Delta\mathbf{x}_j$ consistent with distance d_{ij} must satisfy

$$(\mathbf{x}_i - \mathbf{x}_j)^T(\Delta\mathbf{x}_i - \Delta\mathbf{x}_j) = 0 \tag{4.12}$$

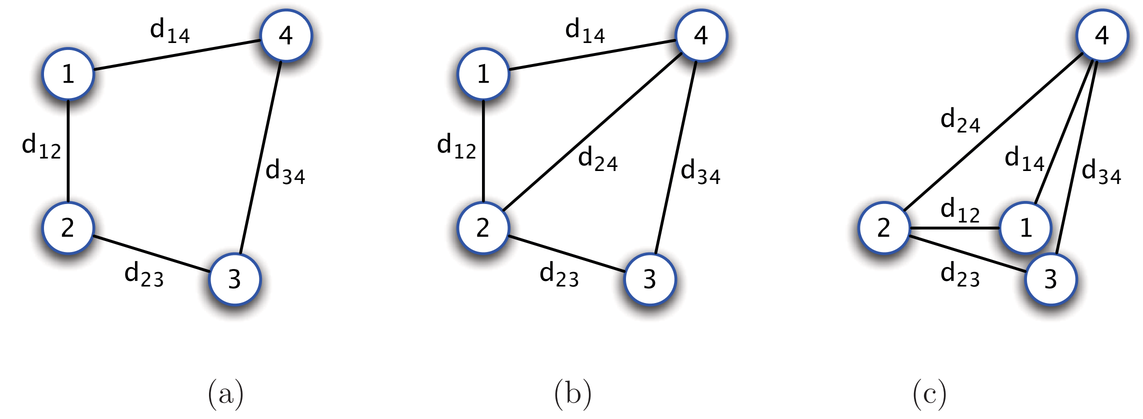

Figure 4.1: (a) Non-rigid formation; (b) first-order rigid, but not globally rigid, formation; (c) first-order rigid configuration having same distances as (b).

This gives as many equations for the local displacements $\{\Delta\mathbf{x}_i\}$ as there are edges in the graph; these may be collected into the linear matrix equation

$$\mathbf{F}\begin{bmatrix}\Delta\mathbf{x}_1\\ \vdots \\ \Delta\mathbf{x}_N\end{bmatrix}, = \mathbf{0} \tag{4.13}$$

exposing a rigidity matrix $\mathbf{F}$ of dimensions $M \times 2N$, where M is the number of edges in the graph. Note that $\mathbf{F}$ has a right nullspace of dimension at least 3, since any displacement in the plane (two degrees of freedom) or rotation (an additional degree of freedom) will not affect the inter-node distances. If all solutions for the displacements $\{\Delta\mathbf{x}_i\}$ are confined to this three-dimensional subspace, the graph is first-order rigid. This is equivalent [41], [42] to the rigidity matrix $\mathbf{F}$ having rank $2N - 3$.

Figure 4.1(a) shows a configuration which is not rigid, since the nodes of the rhombus can move continuously with respect to each other while maintaining the same distances. Either configuration of figure 4.1(b) or (c), however, is first-order rigid. The figure also illustrates, though, that nonuniqueness can still result, since d_{13} admits two solutions consistent with the remaining distances for this example [15].

The stronger notion of *global rigidity* implies that there is a unique configuration consistent with the given distances, modulo the indeterminacies of translation, rotation and reflection, and is characterized (in $\Re^2$) in [43]. To this end, we recall that a graph is *redundantly rigid* if, upon removing an arbitrary edge, it remains first-order rigid. We recall also that a graph is k-connected if, upon removing any set of fewer than k edges, the resulting pruned graph remains connected. We then have:

Theorem 4.1 *([43]). A graph in $\Re^2$ with $N \geq 4$ vertices is globally rigid if and only if it is redundantly rigid and 3-connected.*

Remark: Due to the redundant rigidity constraint, necessarily $2N-2$ distance measurements (or more) must be available: Fewer than this would imply that, upon removing one edge, fewer than $2N-3$ distance measurements would remain, which is less than the number of degrees of freedom in the problem. $\diamond$

The examples in sections 4.4, 4.5 and 4.7 will appeal to this result to ensure that the distance reconstruction problem admits a unique solution; a specific example satisfying theorem 4.1 is the 7-node "camembert" configuration of Figure 4.2. We should note, finally, that uniqueness in low-rank matrix completion has also been addressed by Candès and co-workers [44], [45] who show that, with high probability, a unique solution to the low-rank matrix completion problem exists provided we collect

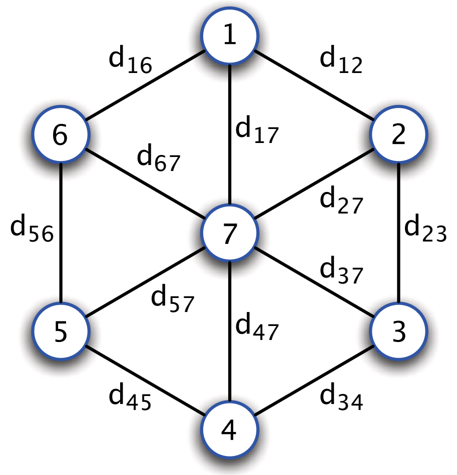

Figure 4.2: A globally rigid formation having the minimum number of edges.

$\mathcal{O}(N\text{poly}(\log N))$ known entries, assuming the singular vectors of the matrix satisfy specific incoherence properties. The precise coefficients in the number of needed entries is not directly revealed, however, which reflects perhaps the generality of the result, as opposed to the more precise graph-theoretic characterization of theorem 4.1, which is clearly more pertinent to the structured problem at hand.

4.4 A Toy Problem

We first an explicit solution to a “toy” low-rank matrix completion problem. Although this approach has some utility when a “small” number of distance measurements are absent, it serves here to establish that the $(1, 3)$-inertia property is necessary, but not sufficient, for a set of squared distances to be consistent. It will likewise illustrate in Section 4.5.1 how nuclear norm minimization [44], [45], [46] may fail.

Suppose for sensor i there is some j for which d_{ij} is unknown or unreliable, but that both sensors i and j can identify three other sensors, with indices l_1, l_2, and l_3, having reliable distance measurements between themselves and with i and j. Without loss of generality, we suppose the indices are numbered as $l_1 = 1$, $l_2 = 2$, $l_3 = 3$, $i = 4$ and $j = 5$. The 5×5 principal submatrix of $\mathbf{D}$ then appears as

$$\mathbf{D}_5 = \begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 & d_{14}^2 & d_{15}^2 \\ d_{21}^2 & 0 & d_{23}^2 & d_{24}^2 & d_{25}^2 \\ d_{31}^2 & d_{32}^2 & 0 & d_{34}^2 & d_{35}^2 \\ d_{41}^2 & d_{42}^2 & d_{43}^2 & 0 & \gamma \\ d_{51}^2 & d_{52}^2 & d_{53}^2 & \gamma & 0 \end{bmatrix} \tag{4.14}$$

where γ denotes the uncertain entry. Since this configuration is globally rigid, the unknown γ is uniquely determined.

One approach is to seek γ such that $\mathbf{D}_5$ is singular, so that its rank cannot exceed four. To facilitate the description to follow, we partition $\mathbf{D}_5$ as:

$$\mathbf{D}_5 = \begin{bmatrix} \mathbf{D}_3 & \mathbf{a} & \mathbf{b} \\ \mathbf{a}^T & 0 & \gamma \\ \mathbf{b}^T & \gamma & 0 \end{bmatrix} \tag{4.15}$$

with $\mathbf{D}_3 \in \Re^{3\times 3}$ and $\mathbf{a}, \mathbf{b} \in \Re^{3\times 1}$. The following lemma proves useful to the main result of this section in Theorem 4.2 below:

Lemma 4.1 *Assume the known distance entries of* $\mathbf{D}_5$ *are consistent and that* $\mathbf{D}_3$

is invertible (i.e., sensor locations 1, 2, and 3 are distinct[2]*). Then*

$$\mathbf{a}^T \mathbf{D}_3^{-1} \mathbf{a} > 0 \qquad \textit{and} \qquad \mathbf{b}^T \mathbf{D}_3^{-1} \mathbf{b} > 0.$$

Proof: Consider

$$\mathbf{D}_4 = \begin{bmatrix} \mathbf{D}_3 & \mathbf{a} \\ \mathbf{a}^T & 0 \end{bmatrix} \tag{4.16}$$

and introduce the indefinite quadratic form

$$f(\mathbf{v}) = \begin{bmatrix} \mathbf{v}^T & 1 \end{bmatrix} \mathbf{D}_4 \begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix} \tag{4.17}$$

where $\mathbf{v} \in \Re^{3\times 1}$ is a free vector. A critical point $\mathbf{v}_*$ of this functional gives a saddle point, obtained where

$$\left.\frac{\partial f(\mathbf{v})}{\partial \mathbf{v}}\right|_{\mathbf{v}=\mathbf{v}_*} = 0, \tag{4.18}$$

yielding $\mathbf{v}_* = -\mathbf{D}_3^{-1}\mathbf{a}$ and $f(\mathbf{v}_*) = -\mathbf{a}^T\mathbf{D}_3^{-1}\mathbf{a}$. The equations at a critical point combine to give

$$\underbrace{\begin{bmatrix} \mathbf{D}_3 & \mathbf{a} \\ \mathbf{a}^T & 0 \end{bmatrix}}_{\mathbf{D}_4} \begin{bmatrix} \mathbf{v}_* \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -\mathbf{a}^T\mathbf{D}_3^{-1}\mathbf{a} \end{bmatrix} \tag{4.19}$$

or

$$\mathbf{a}^T\mathbf{D}_3^{-1}\mathbf{a} = \frac{-1}{(\mathbf{D}_4^{-1})_{44}} \tag{4.20}$$

involving the lower right entry of the inverse $\mathbf{D}_4^{-1}$. It suffices to show that this entry is negative. We appeal to the determinant inverse formula:

$$(\mathbf{D}_4^{-1})_{ij} = \frac{(-1)^{i+j}\det \mathbf{D}_4^{ij}}{\det \mathbf{D}_4} \tag{4.21}$$

[2]Observe that $\det \mathbf{D}_3 = 2\, d_{12}^2\, d_{13}^2\, d_{23}^2$, which is nonzero for distinct locations.

in which $\mathbf{D}_4^{ij}$ is the cofactor which results by deleting row i and column j. A direct calculation shows that

$$\det \mathbf{D}_4^{44} = \det \mathbf{D}_3 = 2\, d_{12}^2\, d_{13}^2\, d_{23}^2 > 0, \tag{4.22}$$

whereas $\det \mathbf{D}_4 < 0$ since this determinant is the product of one positive and three negative eigenvalues, according to Property 4.1. This confirms $\mathbf{a}^T\mathbf{D}_3^{-1}\mathbf{a} > 0$. Permuting $\mathbf{a}$ with $\mathbf{b}$ shows that $\mathbf{b}^T\mathbf{D}_3^{-1}\mathbf{b} > 0$ as well. $\diamond$

Theorem 4.2 *With the same assumptions as Lemma 1:*

1. *The unknown entry is one of two solutions:*

$$\gamma = \mathbf{a}^T\mathbf{D}_3^{-1}(\mathbf{b} \pm \tau\,\mathbf{a}), \tag{4.23}$$

with $\tau = \sqrt{(\mathbf{b}^T\mathbf{D}_3^{-1}\mathbf{b})/(\mathbf{a}^T\mathbf{D}_3^{-1}\mathbf{a})}$.

2. *Both solutions give* $\gamma > 0$ *and result in* $\mathbf{D}_5$ *having inertia* $(1,3)$.

Remark: As only one choice for γ can be correct, this shows that the inertia-$(1,3)$ property (in collaboration with zero diagonal entries and positive off-diagonal terms), while necessary, is not sufficient for a pre-Euclidean distance matrix to be compatible with distance measurements in a plane.

Proof: As $\mathbf{D}_3$ is invertible, we may exploit the block LDU factorization

$$\mathbf{D}_5 = \begin{bmatrix} \mathbf{I}_3 & \\ \begin{bmatrix}\mathbf{a}^T \\ \mathbf{b}^T\end{bmatrix}\mathbf{D}_3^{-1} & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} \mathbf{D}_3 & \\ & \mathbf{D}_5\backslash\mathbf{D}_3 \end{bmatrix} \begin{bmatrix} \mathbf{I}_3 & \mathbf{D}_3^{-1}[\mathbf{a}\ \mathbf{b}] \\ & \mathbf{I}_2 \end{bmatrix} \tag{4.24}$$

which exposes the Schur complement [39], [40], [47] with respect to $\mathbf{D}_3$:

$$\begin{aligned}\mathbf{D}_5 \backslash \mathbf{D}_3 &= \begin{bmatrix} 0 & \gamma \\ \gamma & 0 \end{bmatrix} - \begin{bmatrix} \mathbf{a}^T \\ \mathbf{b}^T \end{bmatrix} \mathbf{D}_3^{-1} \begin{bmatrix} \mathbf{a} & \mathbf{b} \end{bmatrix} \\ &= \begin{bmatrix} -\mathbf{a}^T \mathbf{D}_3^{-1} \mathbf{a} & \gamma - \mathbf{a}^T \mathbf{D}_3^{-1} \mathbf{b} \\ \gamma - \mathbf{b}^T \mathbf{D}_3^{-1} \mathbf{a} & -\mathbf{b}^T \mathbf{D}_3^{-1} \mathbf{b} \end{bmatrix} \end{aligned} \tag{4.25}$$

As $\mathbf{D}_3$ has full rank, $\mathbf{D}_5$ is singular if and only if its Schur complement $\mathbf{D}_5 \backslash \mathbf{D}_3$ is. The determinant $\det \mathbf{D}_5 \backslash \mathbf{D}_3$ gives a quadratic function in γ, whose roots yield part 1 of the theorem statement.

For the second part, we recall that Sylvester's law of inertia [39] asserts that a matrix congruence transformation preserves inertia. Thus the inertia of $\mathbf{D}_5$ is that of $\mathbf{D}_3$ plus that of its Schur complement $\mathbf{D}_5 \backslash \mathbf{D}_3$. As the inertia of $\mathbf{D}_3$ is $(1, 2)$, it suffices to show that the Schur complement $\mathbf{D}_5 \backslash \mathbf{D}_3$ is negative definite for all γ between the critical values isolated in part 1, since it will then have a sole negative eigenvalue when γ reaches either critical value.

To this end, the characteristic polynomial $p(\lambda) = \det(\lambda \mathbf{I}_2 - \mathbf{D}_5 \backslash \mathbf{D}_3)$ becomes

$$p(\lambda) = \lambda^2 + (\alpha + \beta)\,\lambda + \alpha\beta - (\zeta - \gamma)^2 \tag{4.26}$$

with $\alpha = \mathbf{a}^T \mathbf{D}_3^{-1} \mathbf{a}$, $\beta = \mathbf{b}^T \mathbf{D}_3^{-1} \mathbf{b}$ and $\zeta = \mathbf{a}^T \mathbf{D}_3^{-1} \mathbf{b}$. We observe that both α and β are positive from Lemma 4.1, and that the final coefficient $\alpha\beta - (\zeta - \gamma)^2$ is concave in γ. This final coefficient (as a function of γ) must thus be positive between its zero crossings that give the critical values of part 1. This shows that $p(\lambda)$ is a Hurwitz polynomial for values of γ between its critical values, and hence its roots lie in the left-half plane. This confirms that $\mathbf{D}_5\backslash\mathbf{D}_3$ is negative definite for all γ between the critical values, to complete the proof. $\diamond$

To identify which γ from the two choices is correct, we construct $\mathbf{B}$ having entries [cf. (4.6)]

$$\mathbf{B}_{ij} = d_{i5}^2 + d_{j5}^2 - d_{ij}^2, \qquad 1 \le i, j \le 4, \tag{4.27}$$

using $d_{45}^2 = d_{54}^2 = \gamma$. The correct γ of the two choices is identified by $\mathbf{B}$ being positive semidefinite of rank 2. When k pairs of distance measurements are absent, the basic procedure can be applied k times to subsets of 5 sensors.

4.5 Previous Reconstruction Algorithms

We review some existing reconstruction algorithms here, namely nuclear norm minimization [44], [45], [46] and a truncated singular value decomposition approach [16]. Either method may display specific shortcomings, which will motivate the novel algorithms of section 4.6.

4.5.1 Nuclear Norm Minimization

We recall that the nuclear norm $\|\cdot\|_*$ of any matrix is the sum of its singular values:

$$\|\mathbf{D}\|_* = \sum_k \sigma_k \tag{4.28}$$

This norm has particular significance as the convex hull of the rank function of a matrix, when restricted to matrices having induced operator norm (i.e., largest singular value) bounded by one [46]. One may thus approach the minimum rank problem

$$\widehat{\mathbf{D}} = \arg \min_{\mathbf{E} \in \Re^{N \times N}} \operatorname{rank}(\mathbf{E}) \qquad \text{subject to } \mathbf{E}_{ij} = d_{ij}^2 \text{ for all } (i,j) \in \Omega$$

through its relaxation

$$\widehat{\mathbf{D}} = \arg \min_{\mathbf{E} \in \Re^{N \times N}} \|\mathbf{E}\|_* \qquad \text{subject to } \mathbf{E}_{ij} = d_{ij}^2 \text{ for all } (i,j) \in \Omega,$$

as this reduces to minimizing a convex function $\|\mathbf{E}\|_*$ over a convex set.

As Theorem 4.2 shows, a minimum rank solution need not be unique, which is further illustrated in the following example:

Example 1. Consider five sensors located at

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \mathbf{x}_4^T \\ \mathbf{x}_5^T \end{bmatrix} = \begin{bmatrix} 1.17 & -1.50 \\ -0.90 & 0.50 \\ 0.30 & -0.80 \\ 2.00 & 0 \\ 0 & 1.20 \end{bmatrix}$$

As in Theorem 4.1, let $d_{45} = d_{54}$ be the uncertain entry in $\mathbf{D}_5$, with the other entries known. The minimum rank problem admits two solutions according to Theorem 4.1, at

$$\gamma = 5.44, \qquad \gamma = 12.108$$

with the correct solution at $d_{45}^2 = 5.44$. The nuclear norm $\|\mathbf{D}_5\|_*$ (as a function of γ) is minimized at $\gamma = 5.44$, yielding the correct solution in this case.

If instead we consider

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \mathbf{x}_4^T \\ \mathbf{x}_5^T \end{bmatrix} = \begin{bmatrix} -0.15 & 1.50 \\ -1.15 & -0.32 \\ 0 & 0.15 \\ 0.40 & -1.30 \\ -0.60 & 0.60 \end{bmatrix}$$

then the two solutions for γ become

$$\gamma = 0.3852, \qquad \gamma = 4.61$$

with the latter being the correct d_{45}^2. The nuclear norm $\|\mathbf{D}_5\|_*$ is now minimized at $\gamma = 0.3852$ which indeed gives a minimum rank solution, but the incorrect distance reconstruction.

Finally, by considering

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \mathbf{x}_4^T \\ \mathbf{x}_5^T \end{bmatrix} = \begin{bmatrix} 1.00 & -0.50 \\ -0.50 & 0.30 \\ -0.25 & 0.55 \\ -1.50 & 2.50 \\ 0.62 & 0 \end{bmatrix}$$

the minimum rank solutions become

$$\gamma = 3.8735, \qquad \gamma = 10.744$$

whereas the nuclear norm is minimized at $\gamma \approx 1.7287$, which matches nether minimum rank solution. This illustrates how the nuclear norm may not capture the correct solution, even when a well-defined solution exists. $\diamond$

4.5.2 SVD Approximation

The general setting of [16] assumes that we can assess a probability p_{ij} in measuring the distance between sensors i and j. Thus we may construct an incomplete distance matrix with

$$\widehat{\mathbf{D}}_{ij} = \begin{cases} d_{ij}^2, & \text{with probability } p_{ij}; \\ \\ * & \text{with probability } 1 - p_{ij}; \end{cases} \tag{4.29}$$

in which $*$ denotes an unknown value.

A "best guess" estimate of the true matrix of squared distances is then taken as [16]

$$\mathbf{E_{ij}} = \begin{cases} \dfrac{d_{ij}^2 - \gamma_{ij}(1 - p_{ij})}{p_{ij}} & \text{if } (i,j) \in \Omega; \\ \\ \gamma_{ij} & \text{if } (i,j) \in \overline{\Omega}. \end{cases} \tag{4.30}$$

Here the value γ_{ij} is introduced to represent the unknown distance measurement between node i and j. This "guess" can be arbitrary, such as 0, but a good choice does make the reconstruction more accurate. A singular value decomposition is then used to construct the best rank-4 approximant to $\mathbf{E}$ (with this approximant denoted $\mathbf{S}_4$ in [16]), by retaining the first four principal components. It is proved [16] that this procedure provides an unbiased estimate of the true $\mathbf{D}$, and that the distance to the true $\mathbf{D}$ is bounded. To illustrate a shortcoming of this approach, consider a network of seven sensors with locations at

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \mathbf{x}_4^T \\ \mathbf{x}_5^T \\ \mathbf{x}_6^T \\ \mathbf{x}_7^T \end{bmatrix} = \begin{bmatrix} 0.093479 & 1.933876 \\ -0.211001 & 1.078930 \\ 0.588168 & -0.349577 \\ 0.049666 & 1.104135 \\ 0.277079 & 1.008712 \\ -0.648985 & -0.902661 \\ -0.385248 & 0.488242 \end{bmatrix}.$$

The true matrix of squared distances becomes[3]

$$\mathbf{D} = \begin{bmatrix} 0.000 & 0.824 & 5.459 & 0.690 & 0.890 & 8.597 & 2.319 \\ 0.824 & 0.000 & 2.679 & 0.069 & 0.243 & 4.119 & 0.379 \\ 5.459 & 2.679 & 0.000 & 2.403 & 1.942 & 1.836 & 1.649 \\ 0.690 & 0.069 & 2.403 & 0.000 & 0.061 & 4.515 & 0.568 \\ 0.890 & 0.243 & 1.942 & 0.061 & 0.000 & 4.511 & 0.710 \\ 8.597 & 4.119 & 1.836 & 4.515 & 4.511 & 0.000 & 2.004 \\ 2.319 & 0.379 & 1.649 & 0.568 & 0.710 & 2.004 & 0.000 \end{bmatrix}$$

We remove the distance measurements between sensor pairs 1 and 2 (unknown $d_{12} = d_{21}$), 3 and 4 (unknown $d_{34} = d_{43}$) and 2 and 5 (unknown $d_{25} = d_{52}$).

Using first the method of Drineas *et al.* [16] using $\gamma_{ij} = 0$ for the removed entries and $p_{ij} = 1$ for the known entries (which accurately reflects the set-up here),

[3]The displayed version has been rounded to three positions past the decimal point to fit the column width; the actual computations used the full precision representation of $\mathbf{D}$.

the optimal rank-four approximation to $\mathbf{E}$ from (4.30) becomes

$$\begin{bmatrix} 0.227 & -0.094 & 5.586 & 0.714 & 0.671 & 8.624 & 1.844 \\ -0.094 & -0.131 & 2.581 & 0.049 & 0.159 & 4.075 & 0.835 \\ 5.586 & 2.581 & 0.061 & 0.006 & 1.850 & 1.845 & 1.435 \\ 0.714 & 0.049 & 0.006 & 0.015 & 0.007 & 4.509 & 0.588 \\ 0.671 & 0.159 & 1.850 & 0.007 & 0.275 & 4.517 & 0.917 \\ 8.624 & 4.075 & 1.845 & 4.509 & 4.517 & 0.001 & 1.967 \\ 1.844 & 0.835 & 1.435 & 0.588 & 0.917 & 1.967 & 0.822 \end{bmatrix}$$

The eigenvalues of this matrix are $\lambda = 15.0027$, 1.5111, -3.4670 and -11.7765, giving an inertia of $(2, 2)$. By Property 1, this is inconsistent with the matrix of squared distances for any sensor network. Although inspection shows that the diagonal entries are nonzero, this is easily corrected; more critical is that the value reconstructed for d_{12}^2 (or d_{21}^2) is negative, which is fundamentally inconsistent.

4.6 New Algorithms

We construct new iterative reconstruction algorithms, beginning in section 4.6.1 by appealing to the inertia-$(1, 3)$ property of $\mathbf{D}$, followed in section 4.6.2 by appealing to the inertia-$(2, 0)$ of $\mathbf{B}$, and finally a hybrid of the two in section 4.6.3. Simulation results then compare their performance to reconstruction based on nuclear norm minimization.

4.6.1 Iterative Inertial Approximation

The optimal rank four approximant from above need not retain the $(1,3)$ inertia, indicating a fundamental inconsistency via Property 1.

We can thus consider an "inertial" rank-4 approximant, obtained by keeping the most positive and three most negative eigenpairs from the eigendecomposition. That is, given any estimate $\widehat{\mathbf{D}}$ of the matrix of squared distances, compute its eigendecomposition as

$$\widehat{\mathbf{D}} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T, \tag{4.31}$$

with $\mathbf{\Lambda}$ a diagonal matrix containing the eigenvalues and $\mathbf{V}$ the corresponding eigenvectors. An inertial-$(1,3)$ approximant is naturally obtained as

$$\mathbf{E} = \lambda_+\mathbf{v}_+\mathbf{v}_+^T + \lambda_{1-}\mathbf{v}_{1-}\mathbf{v}_{1-}^T + \lambda_{2-}\mathbf{v}_{2-}\mathbf{v}_{2-}^T + \lambda_{3-}\mathbf{v}_{3-}\mathbf{v}_{3-}^T \tag{4.32}$$

in which λ_+ is the most positive eigenvalue, and λ_{1-}, λ_{2-} and λ_{3-} are the three most negative eigenvalues, each paired with its corresponding eigenvector. Optionally, one may replace λ_+ with $|\lambda_{1-}+\lambda_{2-}+\lambda_{3-}|$ to ensure preservation of the zero trace property in addition to inertia.

By setting $\widehat{\mathbf{D}}_0$ as the matrix containing the measured or known squared distances, and replacing unknown or unreliable entries by γ_{ij} ($= 0$ in the simplest case), an iterative refinement runs as follows:

For $k = 0, 1, 2, \ldots$:

1. Let $\mathbf{E}^{(k)}$ be the inertial-$(1,3)$ approximant to $\widehat{\mathbf{D}}^{(k)}$;

2. Update

$$\widehat{\mathbf{D}}_{ij}^{(k+1)} = \begin{cases} \widehat{\mathbf{D}}_{ij}^{(0)} & \text{if } (i,j) \in \Omega; \\ \\ \mathbf{E}_{ij}^{(k)} & \text{if } (i,j) \in \overline{\Omega}. \end{cases} \tag{4.33}$$

If instead the elements of $\widehat{\mathbf{D}}^{(0)}$ are known with confidence values $0 \leq p_{ij} \leq 1$, then step 2 can be replaced with.

2) Update

$$\widehat{\mathbf{D}}_{ij}^{(k+1)} = p_{ij}\widehat{\mathbf{D}}_{ij}^{(0)} + (1 - p_{ij})\mathbf{E}_{ij}^{(k)} \tag{4.34}$$

.

The previous case is obtained with $p_{ij} = 1$ for $(i,j) \in \Omega$ and $p_{ij} = 0$ for $(i,j) \in \overline{\Omega}$.

4.6.2 Inertia $(2,0)$ Reconstruction

A reconstruction exploiting the inertia-$(2,0)$ character of $\mathbf{B}$ from (4.6) may aso be developed, as we pursue here. Suppose the position N is fully connected, i.e., we have d_{iN} for all $i < N$. By partitioning the distance matrix as

$$\mathbf{D}_N = \begin{bmatrix} \mathbf{D}_{N-1} & \mathbf{d} \\ \\ \mathbf{d}^T & 0 \end{bmatrix} \tag{4.35}$$

with $\mathbf{d}^T = [d_{1N}^2, \ldots, d_{N-1,N}^2]$, the matrix $\mathbf{B}$ from (4.6) becomes

$$\mathbf{B} = \mathbf{d}\mathbf{1}^T + \mathbf{1}\mathbf{d}^T - \mathbf{D}_{N-1}, \tag{4.36}$$

where $\mathbf{1}$ now denotes the vector of ones of dimension $N-1$. This matrix has inertia $(2,0)$ if and only if $\mathbf{D}_N$ is a Euclidean distance matrix of dimension 2. This suggests the following iterative algorithm:

For $k = 0, 1, 2, \ldots,$

1. Set

$$\hat{\mathbf{B}}^{(k+1)} = \mathbf{d}\mathbf{1}^T + \mathbf{1}\mathbf{d}^T - \hat{\mathbf{D}}^{(k)}_{N-1} \tag{4.37}$$

2. Calculate the optimal inertia-$(2,0)$ approximant

$$\mathbf{C}^{(k+1)} = \lambda_1 \mathbf{v}_1 \mathbf{v}_1^T + \lambda_2\, \mathbf{v}_2\, \mathbf{v}_2^T \tag{4.38}$$

   where λ_1 and λ_2 are the two most positive eigenvalues of $\hat{\mathbf{B}}^{(k+1)}$, with $\mathbf{v}_1$ and $\mathbf{v}_2$ the corresponding eigenvectors.

3. Update

$$[\widehat{\mathbf{D}}^{(k+1)}_{N-1}]_{ij} = \begin{cases} \widehat{\mathbf{D}}^{(0)}_{ij} & \text{if } (i,j) \in \Omega; \\ [\mathbf{d}\mathbf{1}^T + \mathbf{1}\mathbf{d}^T - \mathbf{C}^{(k+1)}]_{ij} & \text{if } (i,j) \in \overline{\Omega}. \end{cases} \tag{4.39}$$

As before, if instead the elements of $\widehat{\mathbf{D}}^{(0)}$ are known with confidence values p_{ij}, step 3 can be replaced with

3) Update

$$[\widehat{\mathbf{D}}^{(k+1)}_{N-1}]_{ij} = p_{ij}\, [\widehat{\mathbf{D}}^{(0)}]_{ij} + (1 - p_{ij})\, [\mathbf{d}\mathbf{1}^T + \mathbf{1}\mathbf{d}^T - \mathbf{C}^{(k+1)}]_{ij} \tag{4.40}$$

.

Note that the entries d_{iN}^2 are never altered, which impedes application of this algorithm if no node proves fully connected. One workaround is to permute the indices of the nodes at each iteration, in such a way that all unknown entries of $\mathbf{D}_N$ are ultimately updated. An alternate approach is to combine both inertial approximation algorithms into a hybrid, as developed next.

4.6.3 A Hybrid Algorithm

The inertia-(2,0) algorithm assumes that at least one sensor is fully connected (indexed as N by permuting indices, if necessary), whereas the the inertia-(1-3) algorithm need not appeal to such an assumption. We may thus consider a hybrid algorithm that alternately projects between the two approaches, as follows:

For $k = 0,\, 1,\, 2,\, \ldots,$

1. (Inertia-$(1,3)$ projection)

$$\mathbf{D}^+_{ij} = \begin{cases} \widehat{\mathbf{D}}^{(0)}_{ij}, & \text{if } (i,j) \in \Omega; \\ \mathbf{E}^{(k)}_{ij}, & \text{if } (i,j) \in \overline{\Omega}. \end{cases} \tag{4.41}$$

   with $\mathbf{E}^{(k)}$ the inertia-$(1,3)$ approximant to $\mathbf{D}^{(k)}$.

2. (Inertia-$(2,0)$ projection) Partition

$$\mathbf{D}^+ = \begin{bmatrix} \mathbf{D}^{(k)}_{N-1} & \mathbf{d}^{(k+1)} \\ (\mathbf{d}^{(k+1)})^T & 0 \end{bmatrix} \tag{4.42}$$

   and set

$$\hat{\mathbf{B}}^{(k)} = \mathbf{d}^{(k+1)}\mathbf{1}^T + \mathbf{1}(\mathbf{d}^{(k+1)})^T - \mathbf{D}^{(k)}_{N-1} \tag{4.43}$$

Let

$$\mathbf{C}^{(k+1)} = \lambda_1 \mathbf{v}_1 \mathbf{v}_1^T + \lambda_2\, \mathbf{v}_2\, \mathbf{v}_2^T \tag{4.44}$$

be its inertia-$(2, 0)$ approximant.

3. Update

$$[\widehat{\mathbf{D}}_{N-1}^{(k+1)}]_{ij} = \begin{cases} \widehat{\mathbf{D}}_{ij}^{(0)} & \text{if } (i,j) \in \Omega; \\ [\mathbf{d}^{(k+1)}\mathbf{1}^T + \mathbf{1}(\mathbf{d}^{(k+1)})^T - \mathbf{C}^{(k+1)}]_{ij} & \text{if } (i,j) \in \overline{\Omega}. \end{cases} \tag{4.45}$$

and

$$\widehat{\mathbf{D}}_N^{(k+1)} = \begin{bmatrix} \widehat{\mathbf{D}}_{N-1}^{(k+1)} & \mathbf{d}^{(k+1)} \\ (\mathbf{d}^{(k+1)})^T & 0 \end{bmatrix} \tag{4.46}$$

4.7 Simulations

As a first example, we consider the sensor locations from (**??**), while removing the distance measurements between sensor pairs 1 and 2 (unknown $d_{12} = d_{21}$), 3 and 4 (unknown $d_{34} = d_{43}$) and 2 and 5 (unknown $d_{25} = d_{52}$). The unknown values are initialized to zeros in $\hat{\mathbf{D}}^{(0)}$, and figure 4.3 shows the relative error

$$\text{Relative error} = \left(\sum_{i,j} (d_{ij}^2 - \hat{d}_{ij}^2) \right) \Big/ \sum_{i,j} d_{ij}^2 \tag{4.47}$$

versus the iteration number, for the new algorithms of section 4.6. For comparison, the result using a nuclear norm minimization approach adapted via a subgradient is also plotted. The subgradient requires singular vectors to be calculated at each iteration (e.g., [44], [46]), presenting thus a similar computational complexity to the

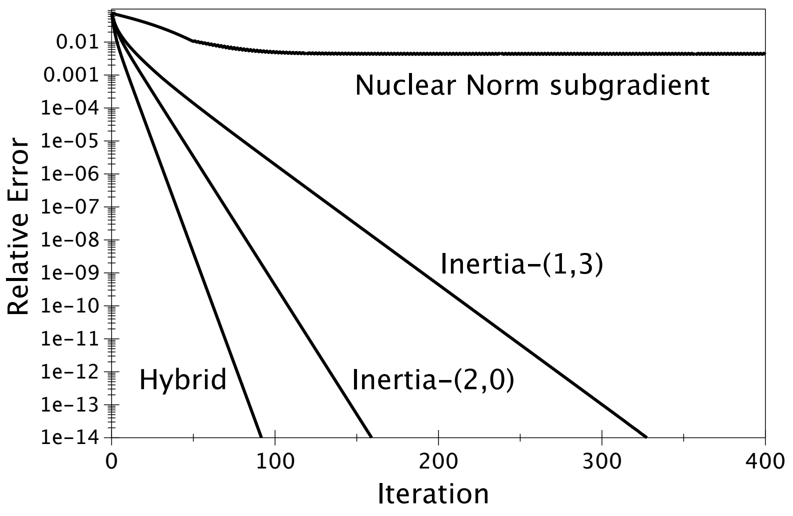

Figure 4.3: Relative error in distance reconstruction for the different algorithms.

algorithms of section 4.6. Although the subgradient algorithm reduces the relative error to comfortably below 1%, it does not offer the same accuracy as the inertia-based algorithms. For this case, the nuclear norm of the reconstructed distance matrix found by the subgradient algorithm is a mere 0.02% greater than the true solution, and further improvements would require searching a subgradient of minimum norm (e.g., [46], [48, §6.3.1]). The alternative algorithm from [49] for nuclear norm minimization did not converge to a useful solution for this example, nor did the SVD approximation approach of [16] (cf. § 4.5.2).

For a second example the sensor locations are

$$\mathbf{X} = \begin{bmatrix} -0.5958 & -0.5018 \\ -1.5225 & -0.1566 \\ 0.3804 & -1.2899 \\ -2.5625 & 0.0863 \\ 0.5802 & -0.7190 \\ -0.3462 & 1.5656 \\ 0.9611 & 0.6097 \end{bmatrix}$$

with the known distances indicated in the connectivity graph of figure 4.4. For this example, the graph is globally rigid (ensuring a unique distance matrix completion), and the number of known entries is at the minimum $2N - 2$ ($= 12$ here), with node 7 fully connected, as with the "camembert" configuration of figure 4.2.

The unknown distances are again initialized to zeros in $\widehat{\mathbf{D}}^{(0)}$, with the relative error in the distance matrix reconstruction for the different algorithms plotted in figure 4.5. The inertia-(2,0) and the hybrid algorithm both succeed for this example. The inertia-(1,3) algorithm has converged to the wrong matrix, although the converged solution is still a pre-Euclidean distance matrix (i.e., a symmetric matrix with zero diagonal and nonnegative elements off the diagonal) of inertia $(1, 3)$. This reflects how the inertia constraint need not ensure that the reconstructed matrix is a Euclidean distance matrix, akin to the examples in section 4.4. The nuclear norm subgradient algorithm also fails for this example, since it has converged to a distance

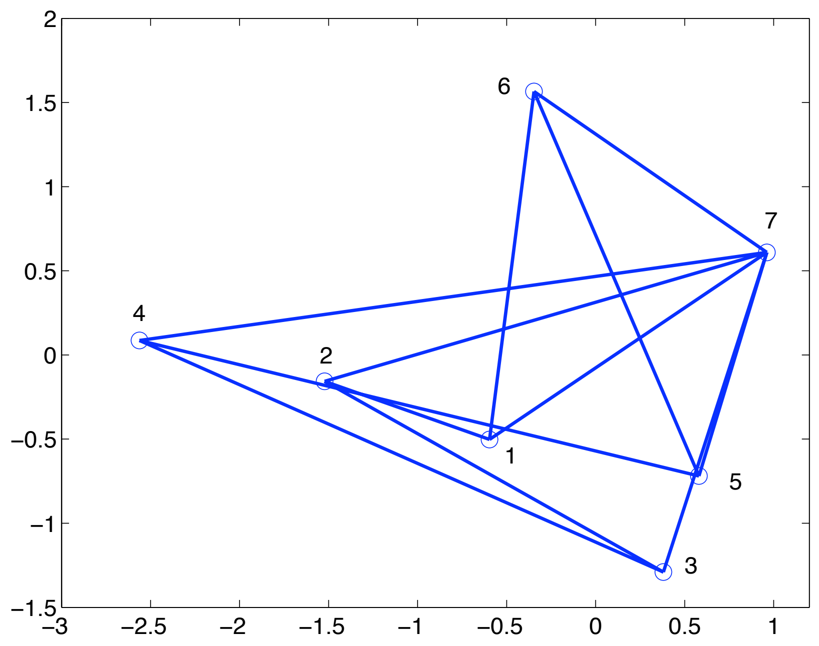

Figure 4.4: Connectivity graph for known distances.

matrix estimate having lower nuclear norm than the true distance matrix, illustrating again how the true distance matrix need not lie at the minimum of the nuclear norm criterion, similar to the examples of section 4.5.1.

4.8 Concluding Remarks

The Euclidean distance matrix reconstruction problem may be cast as a low-rank matrix completion problem, for which considerable algorithmic development as been devoted in recent years [33], [16], [44], [49], [46], [45], [34], . The novelty of the proposed algorithms is the use of inertia, which is more informative than rank: rank is implied from inertia, but not vice-versa. Conversely, inertial constraints do not appear to lend themselves as readily to convex relaxation, and as such the convergence

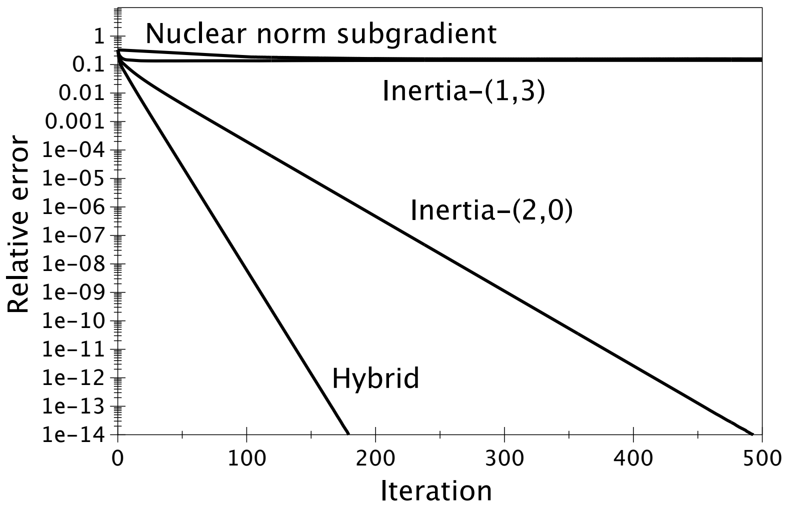

Figure 4.5: Relative reconstruction error for the different algorithms.

properties of the proposed algorithms remain more difficult to establish analytically, despite the favorable behavior observed in examples where convex relaxation methods happen to fall short of expectations. The update procedures of the proposed algorithms, however, display some structural affinities with the subgradient adaptation algorithm, suggesting that some convergence characteristics may yet carry over, as we hope to report in the near future.

Although we have focused on sensor locations in a plane, extensions to three-dimensional space are straightforward: the inertia-$(1,3)$ and inertia-$(2,0)$ constraints are replaced by inertia-$(1,4)$ and inertia-$(3,0)$, respectively. The three-dimensional counterpart to Theorem 4.1, however, is presently unresolved.

Chapter 5

Conclusions And Perspectives For Future Work

Our contributions are that first we developed a wireless sensor network localization algorithm using projections onto the boundary of convex sets. The new method avoids the multimodality problem that plagues least-squares formulations as it has a unique solution. This new method also features a computationally simple update procedure.

Secondly, we provided in this dissertation new algorithms that aim at reconstructing the missing distance measurement information, the information which most localization algorithms are based on. We gave some results on the question of what is the minimum numbers of distance measurements necessary to reconstruct the remaining and if the solution is unique. We found an important feature of the squared distance matrix-inertia, which is more informative than its rank. We developed inertia-(1,3) and inertia-(2,0) approximation reconstruction algorithm. A hybrid algorithm is also given, which applies these two alternatively and converges much faster.

As our research goes on, we met some problems that remain unresolved and will be included in our future work.

We have considered convergence aspect of the two anchor nodes case in the Chapter 3. However, for an arbitrary number of anchor nodes case, we only got some preliminary simulation results. Our goal is to achieve convergence under certain specific situation. One of the possible way to consider this might be find one "global discrepancy function" that converge well in a certain area, or find a particular configuration of the anchor nodes that give one of the "discrepancy function" good conditions to converge.

Another problem in our new projection onto boundary set localization algorithm appears with noisy measurements. One issue is what will the variance of the location estimation after final convergence. Although we can get an answer from numerical simulation, an analytic result is still not available. We will have to further study the mechanism of the boundary projection to determine the variance.

In distance information reconstruction, there remain two problems unresolved. First, the convergence aspect of the Iterative Inertial Approximation. We have not proved yet the convergence aspect of this algorithm. This question are expected to be resolved in the future.

In our distance reconstruction algorithms, noisy distance measurements have not been taken fully into consideration. How sensitive our algorithms are to the noise will be included in our future study too.

Appendix A

Proof of Lemma 1 in Chapter 3

To verify Lemma 1, consider the line ℓ_{ij} passing through anchor nodes $\mathbf{c}_i$ and $\mathbf{c}_j$, parametrized by a scalar α:

$$\ell_{ij} = \Big\{ \mathbf{y} : \mathbf{y} = \mathbf{c}_i + \alpha\,(\mathbf{c}_j - \mathbf{c}_i) \text{ for some } \alpha \Big\}. \tag{A.1}$$

The closest point to $\mathbf{x}$ on this line, denoted $\mathbf{y}_x$, is obtained from the optimization problem

$$\begin{aligned} \mathbf{y}_x &= \arg \min_{\mathbf{y}\,\in\,\ell_{ij}} \|\mathbf{x} - \mathbf{y}\|^2 \\ &= \mathbf{c}_i + \frac{(\mathbf{c}_i - \mathbf{c}_j)\,(\mathbf{c}_i - \mathbf{c}_j)^T}{\|\mathbf{c}_i - \mathbf{c}_j\|^2}\,(\mathbf{x} - \mathbf{c}_i) \end{aligned} \tag{A.2}$$

The difference $\mathbf{x} - \mathbf{y}_x$ gives a line segment normal to ℓ_{ij} (cf. Fig. 3.4):

$$\mathbf{x} - \mathbf{y}_x = \underbrace{\left(\mathbf{I} - \frac{(\mathbf{c}_i - \mathbf{c}_j)\,(\mathbf{c}_i - \mathbf{c}_j)^T}{\|\mathbf{c}_i - \mathbf{c}_j\|^2} \right)}_{\triangleq\,\mathbf{P}_{ij}} (\mathbf{x} - \mathbf{c}_i) \tag{A.3}$$

Now let $\mathbf{z}$ be any other point in the plane. It will lie on the same side of ℓ_{ij} as $\mathbf{x}$ (i.e., $\mathbf{z} \in R_{ij}^{+}$) iff the subspace angle ϕ between the vectors $\mathbf{z} - \mathbf{y}_x$ and $\mathbf{x} - \mathbf{y}_x$ is less than 90 degrees (or $\pi/2$ radians). From the subspace angle formula

$$\cos\phi = \frac{(\mathbf{z} - \mathbf{y}_x)^T (\mathbf{x} - \mathbf{y}_x)}{\|\mathbf{z} - \mathbf{y}_x\|\,\|\mathbf{x} - \mathbf{y}_x\|} \tag{A.4}$$

clearly $|\phi| < \pi/2$ if and only if $(\mathbf{z}-\mathbf{y}_x)^T(\mathbf{x}-\mathbf{y}_x) > 0$. By a straightforward calculation,

$$\begin{aligned}
&(\mathbf{z}-\mathbf{y}_x)^T(\mathbf{x}-\mathbf{y}_x) \\
&= \left(\mathbf{z}-\mathbf{c}_i-\frac{(\mathbf{c}_i-\mathbf{c}_j)\,(\mathbf{c}_i-\mathbf{c}_j)^T}{\|\mathbf{c}_i-\mathbf{c}_j\|^2}(\mathbf{x}-\mathbf{c}_i)\right)^T \mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_i) \\
&= (\mathbf{z}-\mathbf{c}_i)^T\,\mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_i) \qquad \text{(A.5)}
\end{aligned}$$

This final form is thus positive if and only if $\mathbf{z} \in R_{ij}^+$.

We now observe that $\mathbf{P}_{ij}\,(\mathbf{c}_i-\mathbf{c}_j) = \mathbf{0}$ and therefore

$$\mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_i) = \mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_i) + \underbrace{\mathbf{P}_{ij}\,(\mathbf{c}_i-\mathbf{c}_j)}_{0} = \mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_j) \qquad \text{(A.6)}$$

Similarly, $(\mathbf{z}-\mathbf{c}_i)^T\mathbf{P}_{ij} = (\mathbf{z}-\mathbf{c}_j)^T\mathbf{P}_{ij}$. As such, from the update formula

$$\hat{\mathbf{x}}^k - \mathbf{c}_j = d_j\,\frac{\mathbf{x}^{k-1}-\mathbf{c}_j}{\|\mathbf{x}^{k-1}-\mathbf{c}_j\|} \qquad \text{(A.7)}$$

we have

$$\begin{aligned}
&(\hat{\mathbf{x}}^k-\mathbf{c}_j)^T\,\mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_j) \\
&= \frac{d_j}{\|\hat{\mathbf{x}}^{k-1}-\mathbf{c}_j\|}\,(\hat{\mathbf{x}}^{k-1}-\mathbf{c}_j)^T\,\mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_j) \\
&= \frac{d_j}{\|\hat{\mathbf{x}}^{k-1}-\mathbf{c}_j\|}\,(\hat{\mathbf{x}}^{k-1}-\mathbf{c}_i)^T\,\mathbf{P}_{ij}\,(\mathbf{x}-\mathbf{c}_i) \qquad \text{(A.8)}
\end{aligned}$$

As the factor $d_j/\|\hat{\mathbf{x}}^{k-1}-\mathbf{c}_j\|$ is positive, this shows that $\hat{\mathbf{x}}^{k-1}$ and $\hat{\mathbf{x}}^k$ lie in the same half-plane with respect to ℓ_{ij}. ◇

Appendix B

Notes on choosing the missing element

Consider a 5×5 matrix of squared distances

$$\mathbf{D}_5 = \begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 & d_{14}^2 & d_{15}^2 \\ d_{21}^2 & 0 & d_{23}^2 & d_{24}^2 & d_{25}^2 \\ d_{31}^2 & d_{32}^2 & 0 & d_{34}^2 & d_{35}^2 \\ d_{41}^2 & d_{42}^2 & d_{43}^2 & 0 & \star \\ d_{51}^2 & d_{52}^2 & d_{53}^2 & \star & 0 \end{bmatrix} \tag{B.1}$$

in which $\star$ denotes a missing entry that must be filled in, while maintaining rank four. This requires choosing $\star$ such that $\mathbf{D}_5$ is singular, meaning we can find some vector $\begin{bmatrix} \mathbf{z} \\ 1 \end{bmatrix}$ which fulfills

$$\mathbf{D}_5 \begin{bmatrix} \mathbf{z} \\ 1 \end{bmatrix} = \mathbf{0}. \tag{B.2}$$

To facilitate the description to follow, we begin by partitioning $\mathbf{D}_5$ as:

$$\mathbf{D}_5 = \begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 & d_{14}^2 & d_{15}^2 \\ d_{21}^2 & 0 & d_{23}^2 & d_{24}^2 & d_{25}^2 \\ d_{31}^2 & d_{32}^2 & 0 & d_{34}^2 & d_{35}^2 \\ d_{41}^2 & d_{42}^2 & d_{43}^2 & 0 & \star \\ d_{51}^2 & d_{52}^2 & d_{53}^2 & \star & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{D}_3 & \mathbf{a} & \mathbf{b} \\ \mathbf{a}^T & 0 & \star \\ \mathbf{b}^T & \star & 0 \end{bmatrix} \tag{B.3}$$

with $\mathbf{D}_3 \in \Re^{3\times 3}$ and $\mathbf{a}, \mathbf{b} \in \Re^{3\times 1}$.

A means of deducing the element $\star$ may be summarized as follows:

1. Choose
$$\boldsymbol{\xi} = \begin{bmatrix} \mathbf{x} \\ 0 \\ 1 \end{bmatrix} \qquad \text{with } \mathbf{x} \in \Re^{3\times 1}$$
such that
$$\begin{bmatrix} \mathbf{D}_3 & \mathbf{a} & \mathbf{b} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 0 \\ 1 \end{bmatrix} = \mathbf{0} \qquad \Leftrightarrow \qquad \mathbf{x} = -\mathbf{D}_3^{-1}\mathbf{b}.$$

2. Choose a nonzero $\mathbf{y} \in \Re^{4\times 1}$ such that
$$\begin{bmatrix} \mathbf{D}_3 & \mathbf{a} & \mathbf{b} \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ 1 \end{bmatrix} = \mathbf{b} \tag{B.4}$$
This implies that $\mathbf{y}$ is orthogonal to the column space of $\left[\begin{smallmatrix} \mathbf{D}_3 \\ \mathbf{a}^T \end{smallmatrix}\right]$, so that we may take for $\mathbf{y}$ the following form:
$$\mathbf{y} = \left(\mathbf{I}_4 - \begin{bmatrix} \mathbf{D}_3 \\ \mathbf{a}^T \end{bmatrix} \left(\mathbf{D}_3^2 + \mathbf{a}\mathbf{a}^T\right)^{-1} \begin{bmatrix} \mathbf{D}_3 & \mathbf{a} \end{bmatrix} \right) \begin{bmatrix} \mathbf{a} \\ 0 \end{bmatrix} \tag{B.5}$$

3. We now have
$$\begin{bmatrix} \mathbf{D}_3 & \mathbf{a} & \mathbf{b} \end{bmatrix} \left(\begin{bmatrix} \mathbf{x} \\ 0 \\ 1 \end{bmatrix} + \alpha \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix} \right) = \mathbf{b} \tag{B.6}$$

for any α, giving a one-parameter description of all vectors orthogonal to the first three rows of $\mathbf{D}_5$. To have such a vector orthogonal to the fourth row as well, we should have

$$\begin{bmatrix} \mathbf{a}^T & 0 & \star \end{bmatrix} \left(\begin{bmatrix} \mathbf{x} \\ 0 \\ 1 \end{bmatrix} + \alpha \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix} \right) = 0 \tag{B.7}$$

By partitioning

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ y_2 \end{bmatrix}, \qquad \text{with } \mathbf{y}_1 \in \Re^{3\times 1},$$

this simplifies to

$$\star = -\mathbf{a}^T \big(\mathbf{x} + \alpha \mathbf{y}_1 \big), \tag{B.8}$$

which gives $\star$ in terms of α.

4. To achieve orthogonality with the fifth row of $\mathbf{D}_5$, we must have

$$\begin{bmatrix} \mathbf{b}^T & \star & 0 \end{bmatrix} \left(\begin{bmatrix} \mathbf{x} \\ 0 \\ 1 \end{bmatrix} + \alpha \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix} \right) = 0 \tag{B.9}$$

or

$$\mathbf{b}^T (\mathbf{x} + \alpha \mathbf{y}_1) + \alpha y_2 \star = 0 \tag{B.10}$$

giving a second equation connecting α and $\star$:

$$\star = -\frac{\mathbf{b}^T (\mathbf{x} + \alpha \mathbf{y}_1)}{\alpha y_2} \tag{B.11}$$

5. By equating (B.8) and (B.11), we have

$$\mathbf{a}^T\big(\mathbf{x} + \alpha \mathbf{y}_1\big) = \frac{\mathbf{b}^T(\mathbf{x} + \alpha \mathbf{y}_1)}{\alpha y_2} \tag{B.12}$$

which allows us to obtain α as a root of the second-order polynomial

$$(\mathbf{a}^T\mathbf{y}_1\, y_2)\alpha^2 + (\mathbf{a}^T\mathbf{x}\, y_2 - \mathbf{b}^T\mathbf{y}_1)\alpha - \mathbf{b}^T\mathbf{x} = 0. \tag{B.13}$$

One of these roots gives $\star$ via either (B.8) or (B.11).

Appendix C

Multidimensional Scaling

Multidimensional Scaling (or MDS) is a set of mathematical techniques that enable a researcher to uncover the "hidden structure" of data [50]. More reference include [19],[17],[18]can be find as good text for researchers. The first procedure for MDS is based on the theorem by Young and Householder (1938),which gives a method for constructing the configuration given (Euclidean) distances among the points.

[35] have given the necessary and sufficient conditions for a set of numbers to be the mutual distances of a set of real points in Euclidian space, and matrices are found whose ranks determine the dimension of the smallest Euclidean space containing such points. Their theorems involve two basic matrices, $\mathbf{B_i}$ and $\mathbf{F}$.

Let i, j and k be alternate subscripts for n points $(i, j, k = 1, 2, \ldots, n)$ and d_{ij}, d_{ik}, and d_{jk} be the distances between the points, then $\mathbf{B_i}$ is an $(n-1) \times (n-1)$ symmetric matrix with elements

$$b_{jk} = \frac{1}{2}(d_{ij}^2 + d_{ik}^2 - d_{jk}^2) \tag{C.1}$$

The element b_{jk} may be considered to be the scalar product of vectors form point i to points j and k. This follows directly from the cosine law. That is, given the three points i, j and k,

$$d_{jk}^2 = d_{ij}^2 + d_{ik}^2 - 2d_{ij}d_{ik}\cos\theta_{jik} \tag{C.2}$$

which rearranged becomes

$$d_{ij}d_{ik}\cos\theta_{jik} = \frac{1}{2}(d_{ij}^2 + d_{ik}^2 - d_{jk}^2) \tag{C.3}$$

From Equations C.1and C.3, it is seen that $b_{ik} = d_{ij}d_{ik}\cos\theta_{jik}$, the scalar product of vectors point i to points j and k. Matrix $\mathbf{B_i}$ is thus a matrix of scalar products of vectors with origin at point i. There are, n possible $\mathbf{B_i}$ matrices, since i may assume any value from 1 to n.

Matrix $\mathbf{F}$ is an $(n+1)\times(n+1)$ symmetric matrix of squares of inter-point distances bordered by a row and column of ones as follows:

$$\mathbf{F} = \begin{bmatrix} 0 & d_{12}^2 & \dots & d_{1k}^2 & \dots & d_{1n}^2 & 1 \\ d_{21}^2 & 0 & \dots & d_{2k}^2 & \dots & d_{2n}^2 & 1 \\ \vdots & \vdots & & & & \vdots & \vdots \\ d_{i1}^2 & d_{i2}^2 & \dots & d_{ik}^2 & \dots & d_{in}^2 & 1 \\ \vdots & \vdots & & & & \vdots & \vdots \\ d_{n1}^2 & d_{n2}^2 & \dots & d_{nk}^2 & \dots & 0 & 1 \\ 1 & 1 & \dots & 1 & \dots & 1 & 0 \end{bmatrix} \tag{C.4}$$

Young and Householder have shown that:

1. If any matrix $\mathbf{B_i}$ is positive semidefinite, the distances may be considered to be

the distances between points lying in a real Euclidean space.

2. The rank of any positive semidefinite matrix $\mathbf{B_i}$ is equal to the dimensionality of the set of points.

3. The rank of matrix $\mathbf{F}$ is two greater than the dimensionality of the set of points.

4. Any positive semidefinite matrix $\mathbf{B_i}$ may be factored to obtain a matrix $\mathbf{A_i}$ such that

$$\mathbf{B_i} = \mathbf{A_i}\mathbf{A_i'} \tag{C.5}$$

If the rank of $\mathbf{B_i}$ is r, where $r \leq (n-1)$, then matrix $\mathbf{A_i}$ is an $(n-1) \times r$ matrix of projections of points on r orthogonal axes with origin at the i-th point of the r-dimensional, real Euclidean space.

Appendix D

Matlab Code

D.1 Projection onto boundary three points test

```
K=1;

dither = rand * 2 * pi / K;

% Check iterations to see if distance from three points can cause

% a fourth to converge.

% three points (anchors)

x1=[-1;-0.866];

x2=[1;-0.866];

x3=[0;0.866];

% choose random "true" point

%

%y1 = randn(2,1);

%

% plot them

%

figure(1)
```

```
plot(x1(1), x1(2), 'g+');

axis([-5 5 -5 5])

axis('square');

hold on;

plot(x2(1), x2(2), 'g+');

plot(x3(1), x3(2), 'g+');

%

% get true distances

%

%d4 = norm(x4 - y1);

%

% plot circles

%

y1=[0.8898;-2.6217];

plot(y1(1), y1(2), 'r+');

d1 = norm(x1 - y1);

d2 = norm(x2 - y1);

d3 = norm(x3 - y1);

tt = [0:0.01:2]*pi;

plot(x1(1) + d1 * cos(tt), x1(2) + d1 * sin(tt), '-.');

plot(x2(1) + d2 * cos(tt), x2(2) + d2 * sin(tt), '-.');
```

```
    plot(x3(1) + d3 * cos(tt), x3(2) + d3 * sin(tt), '-.');

    %

    % Choose random starting point

    %

    for jj=1:K

    z1 = [x1(1) + d1 * cos(dither + 2*jj*pi/K); x1(2) + d1 * sin(dither
+ 2*jj*pi/K)]

    plot(z1(1),z1(2),'b*');

    %

    % and iterate:

    % for ii=1:20

    [u,t] = loc2(d1,x1(1),x1(2),z1(1),z1(2));

    tr2(z1(1),u,z1(2),t);

    z1(1) = u;

    z1(2) = t;

    [u,t] = loc2(d2,x2(1),x2(2),z1(1),z1(2));

    tr2(z1(1),u,z1(2),t);

    z1(1) = u;

    z1(2) = t;

    [u,t] = loc2(d3,x3(1),x3(2),z1(1),z1(2));

    tr2(z1(1),u,z1(2),t);
    z1(1) = u;

    z1(2) = t;

    end

    end

    hold off;

-------------------------------------------------------------------------

function tr2(x1,x2,y1,y2)

    plot([x1 x2],[y1 y2],'-k');
```

D.2 Projection onto boundary using a random sequencing

```
    % updating operations rather than a cyclic one, to break limit

cycles.

    K=40;

    dither = rand * 2 * pi / K;

    % Check iterations to see if distance from three points can cause

    % a fourth to converge.

    % three points (anchors)

    x1=[-1;-0.866];

    x2=[1;-0.866];

    x3=[0;0.866];

    % choose random "true" point
    % plot them

    figure(1)

    plot(x1(1), x1(2),'g+');

    axis([-5 5 -5 5])

    axis('square');

    hold on;

    plot(x2(1), x2(2),'g+');

    plot(x3(1), x3(2),'g+');

    %

    % get true distances

    %

    % plot circles

    %

    y1=[0.8898;-2.6217];

    plot(y1(1), y1(2),'r+');

    d1 = norm(x1 - y1);

    d2 = norm(x2 - y1);

    d3 = norm(x3 - y1);

    tt = [0:0.01:2]*pi;

    plot(x1(1) + d1 * cos(tt), x1(2) + d1 * sin(tt),'-.');

    plot(x2(1) + d2 * cos(tt), x2(2) + d2 * sin(tt),'-.');
```

```
    plot(x3(1) + d3 * cos(tt), x3(2) + d3 * sin(tt),'-.');
    tr(x1(1),x2(1),x1(2),x2(2));
    tr(x1(1),x3(1),x1(2),x3(2));
    tr(x2(1),x3(1),x2(2),x3(2));
    %
    % Choose random starting point
    %
    FinalPoint = zeros(K,2);
    for jj=1:K
    z1 = [x1(1) + d1 * cos(dither + 2*jj*pi/K); x1(2) + d1 * sin(dither
+ 2*jj*pi/K)];
    plot(z1(1),z1(2),'b*');
    %
    % and iterate:
    %
    current_state = 1;
    tworeg = [1 3];
    for ii=1:60
    switch current_state
    case 1
    [u,t] = loc2(d1,x1(1),x1(2),z1(1),z1(2));
```

```
tr2(z1(1),u,z1(2),t);

z1(1) = u;

z1(2) = t;

current_state = round(2.0 + rand);

case 2

[u,t] = loc2(d2,x2(1),x2(2),z1(1),z1(2));

tr2(z1(1),u,z1(2),t);

z1(1) = u;

z1(2) = t;

current_state = tworeg(round(1.0 + rand));

case 3

[u,t] = loc2(d3,x3(1),x3(2),z1(1),z1(2));

tr2(z1(1),u,z1(2),t);

z1(1) = u;

z1(2) = t;

current_state = round(1.0 + rand);

end

end

FinalPoint(jj,:)  = [u t];

end

hold off;
```

```
figure(2);

plot(x1(1) + d1 * cos(tt), x1(2) + d1 * sin(tt),'-.');

axis([-5 5 -5 5])

axis('square');

hold on;

plot(x2(1) + d2 * cos(tt), x2(2) + d2 * sin(tt),'-.');

plot(x3(1) + d3 * cos(tt), x3(2) + d3 * sin(tt),'-.');

plot(FinalPoint(:,1),FinalPoint(:,2),'ro');

hold off;
```

D.3 Linear algebra method of reconstructing missing distance

```
% Test linear algebra method of reconstructing missing distance

%

% Choose five points randomly

%

Xmat = randn(5,2);

%

% Build matrix of squared distances

%

yvec = sum( (Xmat').2̂);

ovec = ones(size(yvec));
```

```
Dmat = ovec' * yvec + yvec' * ovec - 2 * Xmat * Xmat';

%

% remove (4,5) and (5,4) elements

%

Dnew = Dmat;

Dnew(4,5) = 0.0;

Dnew(5,4) = 0.0;

%

% partition elements of matrix

%

D3 = Dmat(1:3,1:3);

avec = Dmat(1:3,4);

bvec = Dmat(1:3,5);

%

% get x

%

xx = -D3\bvec;

%

% get y

%

Auxmat = [D3; avec'];
```

```
    [Q,R] = qr(Auxmat);

    y1 = Q(1:3,4);

    y2 = Q(4,4);

    %

    % get coefficients of polynomial

    %

    coefs = [y2 * (avec' * y1), (y2 * avec' * xx - bvec' * y1), -bvec'

* xx];

    alpha = roots(coefs);

    %

    % get solutions

    %

    sol1 = - avec' * (xx + alpha(1) * y1)

    sol2 = - avec' * (xx + alpha(2) * y1)

    %

    % Compare with true value

    %

    fprintf('Dmat(4,5) =
```

D.4 Comparation of rank 4 and rank 2 approximations

```
    % This compares rank 4 and rank 2 approximations,
    % using randomly plotted data.
    %
    % Number of nodes
    %
    nnodes = 7;
    %
    % true node locations
    %
    xn = randn(nnodes,1) + i * randn(nnodes,1);
    %
    % example from ICASSP paper
    %
    % low res version:
    xn = [0.09; -0.21; 0.59; 0.05; 0.28; -0.65; -0.39] + i * ...
    [1.93; 1.08; -0.35; 1.10; 1.00; -0.90; 0.50];
    % higer res version:
    xn = [0.093479; -0.211001; 0.588168; 0.049666; 0.277079; -0.648985;
-0.385248]
+ i * ...
    [1.933876; 1.078930; -0.349577; 1.104135; 1.008712; -0.902661;
0.488242];
```

```
%
% Make 7th node the centroid of a circle
%
%
% Matrix containing squares of distances
%
ovec = ones(size(xn));
DMat = real((abs( xn * ovec' - ovec * conj(xn') )).^2);
%
% Check rank
%
rd = rank(DMat)
%
% Operation mode:
% 0 = zero out randomly selected positions
% 1 = add gaussian noise
% 2 = 7-node minimal config.
%
mode = 0;
%
% zero out specific measurements
```

```
%

Diter = DMat;

if (mode==1)

Diter = Diter + 0.1 * randn(size(Diter));

pr = 0.95;

qr = 1 - pr;

Diter = 0.5 * (Diter + Diter');

for jj=1:nnodes

Diter(jj,jj) = 0.0;

end

elseif ((mode==2) && (nnodes==7))

dmas = zeros(7,7);

for jj=1:5

dmas(jj,jj+1) = 1;

end

dmas(1,6) = 1;

dmas(1:6,7) = ones(6,1);

dmas = dmas + dmas';

Diter = DMat .* dmas;

else

%
```

```
maxzeros = 4;

already = zeros(2,maxzeros);

count = 0;

while (already(1,maxzeros)==0)

flag = 0;

ipair = round((nnodes-1) * rand(2,1) + 1);

if (ipair(1)  = ipair(2))

ipair = sort(ipair);

if (count>0)

for jj=1:count

if (sum(ipair - already(:,jj))==0)

flag = 1;

end

end

end

if (flag == 0)

count = count+1;

already(:,count) = ipair;

Diter(ipair(1),ipair(2)) = 0.0;

Diter(ipair(2),ipair(1)) = 0.0;

end
```

```
end

end

end

bindec = NucNormOpt(DMat,Diter)

%

% ICASSP example

%

Diter = DMat;

Diter(1,2) = 0.0;

Diter(2,1) = 0.0;

Diter(3,4) = 0.0;

Diter(4,3) = 0.0;

Diter(2,5) = 0.0;

Diter(5,2) = 0.0;

D0 = Diter;

%

% Plot positions and branches indicating known distances

%

figure(1);

plot(real(xn),imag(xn),'x');

hold on;
```

```
for jj=1:nnodes-1

for kk=jj+1:nnodes

if (D0(jj,kk))

plot([real(xn(jj)) real(xn(kk))],[imag(xn(jj)) imag(xn(kk))])

end

end

end

hold off;

%

% make mask

%

mmask = double(Diter == 0.0);

for jj=1:nnodes

mmask(jj,jj) = 0.0;

end

%

% Iterative loop

%

iter = 600;

crit = zeros(1,iter+1);

crit(1) = sum(sum(abs(DMat - Diter)));
```

```
crit2 = crit;

crit3 = crit;

crit4 = crit;

nunorm = zeros(1,iter);

convmeas = zeros(1,iter);

den = sum(sum(abs(DMat)));

smallsvs = zeros(nnodes-4,iter);

%

% vectors/matrices for rank-2 approach

%

vvec = DMat(1:nnodes-1,nnodes);

oovec = ones(nnodes-1,1);

fmat = oovec * vvec' + vvec * oovec';

D2 = D0;

D3 = D0;

D4 = D0;

mmask2 = mmask(1:nnodes-1,1:nnodes-1);

%

% Iterative loop

%

freeze1 = 0;
```

```
freeze3 = 0;

oftcount = 0;

countaber = zeros(5,1);

stepsize = 0.1;

for jj=1:iter

Dold = Diter;

%

% SVD approximation

%

% [U,S,V] = svd(Diter);

% Diter = D0 + mmask .* ( U(:,1:4) * S(1:4,1:4) * (V(:,1:4))');

%

% (1,3)-inertial approximation

%

if (freeze1==0)

[V,Lambda] = eig(Diter);

[ss,pm] = sort(diag(Lambda));

count = 0;

for mm=1:3

if (ss(mm) < 0)

count = count+1;
```

```
    end
    end
    if (count < 3)
    oftcount = oftcount+1;
    countaber(oftcount) = jj;
    end
    recon = V(:,pm(1:count)) * Lambda(pm(1:count),pm(1:count))
* (V(:,pm(1:count)))';
    lval = abs(sum(diag(Lambda(pm(1:3),pm(1:3)))));
    recon = recon + V(:,pm(nnodes)) * lval * (V(:,pm(nnodes)) )';
    if (mode==1)
    Diter = pr * Diter + qr * recon;
    else
    Diter = D0 + mmask .* recon;
    end
    end
    crit(jj+1) = sum(sum(abs(DMat - Diter)));
    if ((crit(jj+1)/den)<1e-15)
    freeze1 = 1;
    end
    %
    % Rank-2 B approx

    %

    Bmat = fmat - D2(1:nnodes-1,1:nnodes-1);

    [U,Mu] = eig(Bmat);

    [tt,pp] = sort(diag(Mu));

    ivec = pp(nnodes-2:nnodes-1);

    recon2 = U(:,ivec) * Mu(ivec,ivec) * (U(:,ivec))';

    Bmat = fmat - recon2;

    if (mode==1)

    D2(1:nnodes-1,1:nnodes-1) = pr * D2(1:nnodes-1,1:nnodes-1)

+ qr * Bmat;

    else

    D2(1:nnodes-1,1:nnodes-1) = D0(1:nnodes-1,1:nnodes-1)

+ mmask2 .* Bmat;

    end

    crit2(jj+1) = sum(sum(abs(DMat - D2)));

    %

    % Composite algorithm

    %

    if (freeze3==0)

    [V,Lambda] = eig(D3);
```

```
     [ss,pm] = sort(diag(Lambda));
     recon = V(:,pm(1:3)) * Lambda(pm(1:3),pm(1:3))
* (V(:,pm(1:3)))';
     lval = abs(sum(diag(Lambda(pm(1:3),pm(1:3)))));
     recon = recon + V(:,pm(nnodes)) * lval * (V(:,pm(nnodes)) )';
     if (mode==1)
     D3 = pr * D3 + qr * recon;
     else
     D3 = D0 + mmask .* recon;
     end
     %
     vvec = D3(1:nnodes-1,nnodes);
     % vvec = recon(1:nnodes-1,nnodes);
     gmat = oovec * vvec' + vvec * oovec';
     Bmat = gmat - D3(1:nnodes-1,1:nnodes-1);
     % Bmat = gmat - recon(1:nnodes-1,1:nnodes-1);
     [U,Mu] = eig(Bmat);
     [tt,pp] = sort(diag(Mu));
     ivec = pp(nnodes-2:nnodes-1);
     recon2 = U(:,ivec) * Mu(ivec,ivec) * (U(:,ivec))';
     Bmat = gmat - recon2;
     if (mode==1)

     D3(1:nnodes-1,1:nnodes-1) =

pr * D3(1:nnodes-1,1:nnodes-1) + qr * Bmat;

     else

     D3(1:nnodes-1,1:nnodes-1) =

D0(1:nnodes-1,1:nnodes-1) + mmask2 .* Bmat;

     end

     end

     crit3(jj+1) = sum(sum(abs(DMat - D3)));

     if ((crit3(jj+1)/den)<1e-15)

     freeze3=1;

     end

     %

     % Nuclear norm minimization

     %

     [U,S,V] = svd(D4);

     % nunorm(jj) = sum(svd(D4));

     nunorm(jj) = sum(diag(S));

     % Sclip = double(S > threshold);

     % subgrad = U * Sclip * V';

     subgrad = U * V';
     % [bindec,subgrad] = NucNormOpt(D4,D0,'quiet');

     D4 = D0 + mmask .* (D4 - stepsize * subgrad);

     D4 = 0.5 * (D4 + D4');

     crit4(jj+1) = sum(sum(abs(DMat - D4)));

     end

     if (oftcount)

     fprintf('The inertia-(1,3) algo failed to find 3

negative eigenvalues

     for jj=1:oftcount

     fprintf('at iteration

     end

     end

     figure(2);

     semilogy([0:iter],crit/den,[0:iter],crit2/den, [0:iter],

crit3/den,[0:iter],crit4/den);

     xlabel('Iteration number');

     ylabel('Relative error in reconstructed distances');

     legend('Inertia (1,3) Approximation','Rank 2 Approximation',

'Composite','Nuclear Norm Minimization','location','southwest');

     figure(3);

     plot(nunorm);
```

```
    truenucnorm = sum(svd(DMat));

    hold on;

    plot([1 iter],[truenucnorm truenucnorm],'r.-')

    xlabel('Iteration number');

    ylabel('Nuclear norm');

    hold off;

    %

    % compare with true nuclear norm

    %

    %fprintf('True nuclear norm =
----------------------------------------------------------------


    function [bindec,subgrad] = NucNormOpt(DMat,Dswiss,quiet)

    %

    % bindec = NucNormOpt(DMat,Dswiss)

    %

    % returns a binary decision whether the first argument DMat is

    % the minimum nuclear norm completion of the punctured matrix

    % given as the secont argument, in which zeros identify the
positions

    % to be filled in.
```

```
%
% check if we want quiet behavior
%
doprint = 1;
if (nargin == 3)
if (quiet == 'quiet')
doprint = 0;
end
end
%
%
% get dimension
%
[mm,nn] = size(DMat);
vv = [1:mm*nn]';
IndMat = reshape(vv,mm,nn);
unvecs = [1; 0];
count = 0;
for jj=1:mm
for kk=1:nn
if (jj  = kk)
if (Dswiss(jj,kk) == 0.0)

count = count+1;

unvecs(count) = IndMat(jj,kk);

end

end

end

end

%

% vec the equation

% assuming DMat is a rank four Euclidean distance matrix

%

[U,S,V] = svd(DMat);

if (doprint)

ss = svd(Dswiss);

fprintf('Nuclear norm before zeroing:


end

Tmat = U(:,1:4) * ( V(:,1:4))';

rhsvec = Tmat(:);

lhsmat = kron(V(:,5:mm),U(:,5:nn));

lilrhs = rhsvec(unvecs);
```

```
     lillhsmat = lhsmat(unvecs,:);

     %

     % Use QR decomposition to get minimum norm solution

     %

     [Qfac,Rtri] = qr(lillhsmat',0);

     sol1 = (Rtri')\lilrhs;

     Wvec = Qfac * sol1;

     Wmat = reshape(Wvec,mm-4,nn-4);

     ss=svd(Wmat);

     if (ss(1) <= 1.0)

     bindec = 1;

     subgrad = U(:,1:4) * (V(:,1:4))' -

U(:,5:mm) * Wmat * (V(:,5:nn))';

     else

     bindec = 0;

     subgrad = U(:,1:4) * (V(:,1:4))' -

U(:,5:mm) * Wmat * (V(:,5:nn))'/ss(1);

     end

     % test zeros

     %

     %zmat = U(:,1:4) * (V(:,1:4))' - U(:,5:mm) * Wmat * (V(:,5:nn))';
```

Bibliography

[1] M.G. Rabbat and R.D. Nowak. Decentralized source localization and tracking. In *Proc. Int. Conf. Acoustics Speech, Signal Processing*, volume III, pages 921–924, Montreal, Canada, May 2004.

[2] Krishna M. Sivalingam C.S.Raghavendra and Taieb Znati. *Wireless Sensor Networks.* Kluwer Academic Publishers, Norwell, MA, 2004.

[3] 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*, pages 54–69, July 2005.

[4] R.L. Moses, D.Krishnamurthy, and R.Patterson. A self-localization method for wireless sensor networks. In *EURASIP J.Applied Sig.Proc.*, pages 348–358, March 2003.

[5] L.Doherty, K.S.J Pister, and L.E. Ghaoui. Convex position estimation in wireless sensor networks. In *IEEE InfoCom*, pages 1655–1663, 2001.

[6] D. Blatt and A. O. Hero. Energy-based sensor network source localization via projection onto convex sets. *IEEE Signal Processing Magazine*, 54(9):3614–3619, 2006.

[7] D.Niculescu and B.Nath. Ad hoc positioning system. In *Proc.IEEE Globecom 2001*, pages 2926–2931, April 2001.

[8] R.Nagpal, H.Shrobe, and J.Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *Proc.2nd Int.Workshop Inform.Proc.in Sensor Networks*, pages 333–348, April 2003.

[9] D.J.Torrieri. Statistical theory of passive location systems. *IEEE Trans.Aerosp.Electron.Syst.,*, AES-20(2):183–198, 1984.

[10] J.Albowicz, A.Chen, and L.Zhang. Recursive position estimation in sensor networks. In *Proc.IEEE Int.Conf.on Network Protocals*, pages 35–41, November 2001.

[11] D.Niculescu and B.Nath. Localized positioning in ad hoc networks. *Elsevier's Journal of Ad Hoc Networks, Special Issue on Sensor Network Protocals and Applicaitons*, 1:247–259, 2003.

[12] C.Savarese, J.Rabay, and K.Langendoen. positioning algorithms for distributed ad-hoc wireless sensor network. In *USENIX Technical Annual Conference*, 2002.

[13] A.Savvides, H.Park, and M.Srivastava. The n-hop multilateration primitive for node localization problems. *ACM Mibile networks and applications*, 8:443–451, 2003.

[14] M. Cao, B. D. O. Anderson, and A. S. Morse. Localization with imprecise distance information in sensor networks. In *Conf. Decision and Control*, pages 1829–1834, December 2005.

[15] J. Aspnes, T. Eren, D. K. Goldenberg, A. S. Morse, W. Whiteley, Y. R. Yang, B. D. O. Anderson, and P. N. Belhumeur. A theory of network localization. *IEEE Trans. Mobile Computing*, 5(12):1663–1678, December 2006.

[16] 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 *Sensor and Ad Hoc Communications and Networks*, volume 2, pages 536–544, September 2006.

[17] W. S. Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, 1952.

[18] W. S. Torgerson. Multidimensional scaling of similarity. *Psychometrika*, 30(4):379–393, 1965.

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

[20] D. Blatt and A. O. Hero. A convergent incremental gradient method with constant step size. In *SIAM J.Optim*, 2004.

[21] D.Li, K.D.Wong, Y.H.Hu, and A.M.Sayeed. Detection classification and tracking of targets. *IEEE Signal Processing Magazine*, 19(2):17–29, March 2002.

[22] X.Sheng and Y.H.Hu. Energy based acoustic localization. In *Information Processing in Sensor Networks, Sencond International Workshop*, California, April 2003.

[23] M.G. Rabbat and R.D. Nowak. Distributed optimization in sensor networks. In *Proceedings of the Tird International Symposium on Information Processing in Sensor Networks*, Berkeley, California, April 2004.

[24] R.D.Nowak. Distributed EM algorithms for density estimation and clustering in sensor networks. *IEEE Trans. Signal Processing*, 51(8):2245–2253, August 2003.

[25] A. O. Hero and D. Blatt. Sensor network source localization via projection onto convex sets (POCS). In *Proc. Int. Conf. Acoustics Speech, Signal Processing*, volume III, pages 689–692, Philadelphia, March 2005.

[26] L.E. Kinsler. *Fundamentals of Acoustics.* John Wiley and Sons, Inc., NY, NY, 1982.

[27] M. Rydstrom, E. G. Strom, and A. Svensson. Robust sensor network positioning based on projection onto circular and hyperbolic convex sets (POCS). In *Proc. SPAWC*, Cannes, France, July 2006.

[28] Chen Meng, Zhi Ding, and Soura Dasgupta. A semidefinite programming approach to source localization in wireless sensor networks. *IEEE Signal Processing Letters*, 15:253–256, 2008.

[29] Y. Shang and W. Ruml. Improved MDS-based localization. In *IEEE InfoCom*, pages 2640–2651, Hong Kong, March 2004.

[30] AO Hero III JA Costa, N Patwari. Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks. *ACM Trans. on Sensor Networks*, 2(1):39–64, 2006.

[31] W. A. Sethares and C. R. Johnson, Jr. A comparison of two quantized state adaptive algorithms. *IEEE Trans. Acoustics, Speech and Signal Processing*, 37(1):138–143, January 1989.

[32] J. Gower. Properties of Euclidean and non-Euclidean distance matrices. *Linear Algebra and its Applications*, 67:81–97, 1985.

[33] A. Y. Alfakih, A. Khandani, and H. Wolkowicz. Solving Euclidean distance matrix completion problems via semidefinite programing. *Computational Optimization and Applications*, 12:13–30, 1999.

[34] Y. Ding, N. Krislock, J. Qian, and H. Wolkowicz. Sensor network localization, Euclidean distance matrix completions, and graph realization. *(u?)*, 2010.

[35] G. Young and A. S. Householder. Discussion of a set of points in terms of their mutual distances. *Psychometrika*, 3(1):19–22, 1938.

[36] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[37] J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. Technical report, Max Planck Institute for Biological Cybernetics, Tübingen, Germany, July 2003.

[38] H. Choi and S. Choi. Kernel isomap. *Electronics Letters*, 40(25):1612–1613, December 2004.

[39] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins Univ. Press, Baltimore, MD, 2nd edition, 1989.

[40] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge Univ. Press, Cambridge, UK, 1985.

[41] B. Roth. Rigid and flexible frameworks. *American Math. Monthly*, 88:6–21, 1981.

[42] W. Whiteley. Rigidity and scene analysis. In J. Goodmand and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 1327–1354. CRC Press, 2004.

[43] B. Jackson and T. Jordán. Connected rigidity matroids and unique realizations of graphs. *J. Combinatorial Theory B*, 94:1–29, 2005.

[44] E. J. Candès and B. Recht. Exact low-rank matrix completion via convex optimization. In *Allerton Conf. Communication, Control and Computing*, pages 806–812, Urbana-Champagne, IL, September 2008.

[45] E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Information Theory*, x(y):1, 2010.

[46] B. Recht, M. Fazel, and P. A. Parillo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52, 2010.

[47] M. Neumann. On the Schur complement and the LU-factorization of a matrix. *Linear and Multilinear Algebra*, 9:241–254, 1981.

[48] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, MA, 2nd edition, 1999.

[49] S. Ma, D. Goldfarb, and L. Chen. Fixed point and Bregman iterative methods for matrix rank minimization. Technical report, Department of IEOR, Columbia University, 2008.

[50] J. B. Kruskal and M. Wish. *Multidimensional Scaling.* Sage, Beverly Hills, CA, 1978.